KARELIA-AMMATTIKORKEAKOULU
Tietotekniikan koulutusohjelma

Jarkko Lösönen

TALOUDEN RAPORTOINNIN KEHITYSTEHTÄVÄ

Opinnäytetyö
Toukokuu 2014

Karelia AMMATTIKORKEAKOULU

OPINNÄYTETYÖ
Toukokuu 2014
Tietotekniikan koulutusohjelma

Karjalankatu 3
80200 JOENSUU
p. (013) 260 6800

Tekijä
Jarkko Lösönen

Nimeke
Talouden raportoinnin kehitystehtävä

Toimeksiantaja
OlapCon Oy

Tiivistelmä

Opinnäytetyön toimeksiantona oli OlapCon Oy:n sisäisen raportoinnin jatkokehitys asiakaskohtaisen katelaskennan osalta. Työn tavoitteena oli saada asiakaskohtainen myynti- ja käyttökatteen laskenta tarpeeksi luotettavalle tasolle, jotta sitä voitiin hyödyntää päivittäisessä päätöksenteossa. Katelaskentaa ulotettiin asiakasnäkökulman lisäksi muihinkin osa-alueisiin, kuten laskutushenkilöihin ja kustannuspaikkoihin.

Työn toteutuksessa tehtiin tietovarasto Microsoft SQL Server 2012:een. Tietovarastossa yhdistettiin tietoa sekä kirjanpidon että toiminnanohjauksen tietokannoista, jotta saatin muodostettua faktataulut, joissa kustannukset ja liikevaihto olivat laskettuna mm. asiakkaittain. Tietovarastoon luotiin myös tarvittavat dimensiotaulut. Tietovaraston pohjalta tehtiin Business Intelligence-ratkaisu, eli tässä tapauksessa OLAP-kuutio IBM Cognos-tuotteilla.

Tuloksena syntyi OLAP-kuutio, jossa liikevaihdon ja kustannusten pohjalta on laskettu mm. asiakaskohtaiset katteet. Kuutio mahdollistaa asiakaskohtaisen katteen tarkastelun ja sen pohjalta yrityksen johto voi tehdä raportointia moniulotteisena analyysina IBM Cognos Analysis Studiolla. Jatkokehitysmahdollisuuksia ovat tietovaraston ja OLAP-kuution laajentaminen, jos tarkasteluun halutaan ottaa muitakin liiketoiminnan osa-alueita. Tietovaraston pohjalta voidaan myös toteuttaa moniulotteisen analyysin lisäksi muitakin Business Intelligence -ratkaisuja, kuten perusraportointia ja dashboardeja.

Kieli
suomi

Sivuja 46

Asiasanat
tietovarastointi, Business Intelligence, OLAP, raportointi

Karelia
UNIVERSITY OF APPLIED SCIENCES

THESIS
May 2014
Degree Programme in
Information Technology
Karjalankatu 3
FI 80200 JOENSUU
FINLAND
Tel. 358–13-260 6800

Author
Jarkko Lösönen

Title
Development Task for Financial Reporting

Commissioned by
OlapCon Oy

Abstract

The assignment for this thesis was to further develop OlapCon Oy's financial reporting to include a customer-specific gross margin. The main objective of the assignment was to get the calculation of the gross margin to a reliable level and allocated to customers so that it could be used as support in making decisions that affect the business. In addition to customers, also other dimensions were included

The first stage of the project was to develop a data warehouse with Microsoft SQL Server 2012 that included fact and dimension tables. Fact tables included calculated revenues and costs allocated to customers and other dimensions. Dimension tables included basic information about the dimensions like customer names. Another stage of the project was to develop a Business Intelligence solution which in this case was an OLAP-cube based on the data warehouse. IBM Cognos products were used in the Business Intelligence process.

The result of this assignment was an OLAP-cube where gross margin was calculated based on the revenue and costs. Gross margin was allocated to customers and other required dimensions in the cube. The management can use the cube as a data source in multidimensional analysis with IBM Cognos Analysis Studio to make different reports. Some further development ideas for this project include expanding the data warehouse and OLAP-cube to include other important measures.

Language

Finnish

Pages 46

Keywords
data warehousing, Business Intelligence, OLAP, reporting

Sisältö

Lyhenneluettelo....5
1 Johdanto....6
2 Tietovarastointi ja Business Intelligence....7
2.1 SQL ja relaatiotietokannat....9
2.2 ETL-prosessi ja tietovarastokehityksen tekniikat....12
2.3 Business Intelligence ja OLAP....15
3 Toteutuksessa käytetyt teknologiat....19
3.1 Microsoft SQL Server 2012....19
3.2 IBM Cognos Framework Manager....20
3.3 IBM Cognos Transformer....21
3.4 IBM Cognos Analysis Studio....22
4 Opinnäytetyön lähtöasetelma ja tavoitteet....22
5 Työn toteutus....23
5.1 Tietovaraston suunnittelu....24
5.2 ETL-prosessi....25
5.3 Framework-paketin toteutus....32
5.4 OLAP-kuution toteutus Transformerilla....35
5.5 Kuution tarkastelu Analysis Studiolla....39
5.6 Työn lopputoimet....41
6 Tulokset....42
7 Pohdinta....43
Lähteet....45

Lyhenneluettelo

BI	Business Intelligence, liiketoimintatiedon käsittelyä, esittämistä ja hyödyntämistä yrityksen päätöksenteossa
CRM	Customer Relationship Management, asiakkuudenhallinta
CSV	Comma-separated Values, tekstitiedosto, jossa taulukon tiedot on eroteltu toisistaan pilkuilla ja rivinvaihdoilla
ERP	Enterprise Resource Planning, toiminnanohjausjärjestelmä
ETL	Extract - Transform – Load, tietovarastointiin liittyvä prosessi, jossa tietoa poimitaan, muokataan ja ladataan
OLAP	Online Analytical Processing, moniulotteinen analysointi
SQL	Structured Query Language, standardoitu kyselykieli
SSAS	SQL Server Analysis Services, Microsoft SQL Serverin OLAP-työkalu
SSIS	SQL Server Integration Services, Microsoft SQL Serverin ETL-työkalu
SSMS	SQL Server Management Studio, Microsoft SQL Serverin hallintatyökalu
SSRS	SQL Server Reporting Services, Microsoft SQL Serverin raportointityökalu
T-SQL	Transact-SQL, Microsoftin kehittämä versio SQL-kyselykielestä

1 Johdanto

Opinnäytetyön aiheena oli toimeksiantajan OlapCon Oy:n talouden raportoinnin jatkokehitys. Toimeksiantajalla oli valmis idea, jonka pohjalta työtä alettiin toteuttamaan. Työ oli toimeksiantajalle tärkeä, sillä talousraportoinnissa katelaskennan jatkokehitys oli ollut jo pitkään suunnitteilla. Työn päätavoitteena oli saada katelaskenta allokoitua asiakkaille, minkä avulla nähtäisiin asiakaskohtaisia tuloksia. Työ toteutettiin toimeksiantajan tiloissa ja käytössä oli yrityksen tietokoneet ja yrityksellä käytössä olevat Business Analytics -teknologiat.

Opinnäytetyön keskeisiä käsitteitä ovat tietovarastointi ja SQL sekä Business Intelligence (BI). Tietovarastoinnissa tietoa kerätään monesta eri paikasta ja jalostetaan sellaiseen muotoon, että siitä saadaan erilaisilla BI-tuotteilla aikaan ratkaisuja, jotka kertovat tärkeistä liiketoiminnan osa-alueista. Tietovarastointi ja BI sisältävät tiedonhankintaa, -käsittelyä, -tallennusta, -analysointia sekä esittämistä. Erilaiset raportit, dashboardit ja analyysit ovat yleisimpiä lopputuloksia tiedon kuvauksessa ja niitä käytetään tukena yrityksen päätöksenteossa.

Business Intelligence on yksi osa OlapConin tarjoamista palveluista ja toimeksiantona toteutetun työn prosessi vastaa Business Intelligence -töitä, joita OlapCon tarjoaa yritysasiakkailleen. Tässä dokumentissa perehdytään tietovarastointiin ja Business Intelligenceen käsitteinä sekä tutkitaan niiden sisältöä ja tarjoamia hyötyjä. Työn toteutuksessa käytettyjä teknologioita ovat tietovarastointiin käytetty Microsoft SQL Server 2012, metatiedon mallinnukseen käytetty IBM Cognos Framework Manager, OLAP-kuution suunnitteluun ja toteutukseen tarkoitettu IBM Cognos Transformer sekä loppuraportoinnin toteuttamiseen moniulotteisena analyysina käytetty IBM Cognos Analysis Studio. Opinnäytetyössä on esitelty perusteet käytetyistä teknologioista sekä niillä syntyvistä tuloksista.

Työn toteuttamiseksi tarvittiin tietovarasto, johon yhdistettiin dataa kirjanpidon ja toiminnanohjausjärjestelmän (ERP-järjestelmä) tietokannoista. Tietovarastointi toteutettiin toimeksiantajan palvelimella käytössä olevalla Microsoft SQL Server 2012:lla. IBM Cognos -tuotteet löytyivät eri palvelimelta ja siellä toteutettiin da-

tan mallinnus ja analysointi. Yhtenä työn tavoitteena oli saada selkeämpi kuva millä menetelmillä asiakaskohtaisen myynti- / käyttökatteen laskenta saadaan tarpeeksi luotettavalle tasolle, jotta saaduilla tuloksilla pystytään ohjaamaan päivittäistä päätöksentekoa entistä tehokkaammin. Katelaskentaa ulotettiin asiakasnäkökulman lisäksi koskemaan myös eri liiketoiminta-alueita. Muita näkökulmia olivat mm. laskutushenkilöt ja kustannuspaikat. Opinnäytetyön tulokset otettiin yrityksessä aktiiviseen käyttöön johtoportaan päätöksenteon tukena. Opinnäytetyössä on otettu huomioon toimeksiantajan vaatimukset siitä, mitä tietoja työn tuloksista dokumentissa saadaan näyttää.

2 Tietovarastointi ja Business Intelligence

Organisaatioissa tieto on yleensä hajallaan eri järjestelmien sisällä. Käytössä voi olla useita erilaisia ja erikokoisia operatiivisia ohjelmistoja (ERP, CRM ym.), jotka kaikki käyttävät omia tietokantojaan. Usein tietoja ei ole myöskään kuvattu, jolloin ei ole tarkkaa tietoa, mitä mikäkin kenttä sisältää. Tietoa voi olla tietokantojen lisäksi myös erilaisissa tiedostoissa ja taulukkolaskentaohjelmissa. Nämä seikat aiheuttavat sen, että tietoa on hyvin vaikeaa hyödyntää nopeasti ja tehokkaasti. [1, s. 4–5.] Tietovarastointi (Data Warehousing) on ratkaisu näiden hajanaisten tietojen hyödyntämiseen. "Tietovaraston päätarkoitus on tarjota helppokäyttöisesti ja nopeasti oikeita vastauksia käyttäjien kyselyihin ja raportteihin" [2, s. 31].

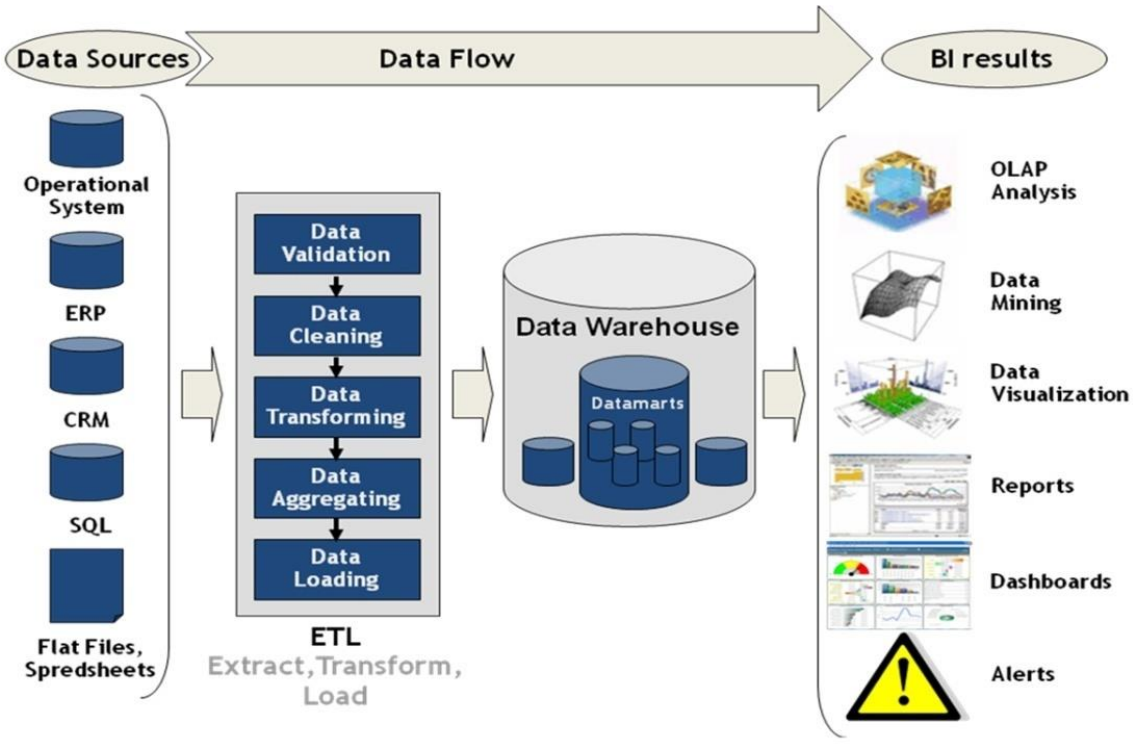
Liiketoiminnan kannalta tietovarastointi parantaa yrityksen kilpailukykyä antamalla mahdollisuuden tiedon analysointiin ja vertailuun. Tietovarastointia sovelletaan tyypillisimmin myynninseurantaan, taloushallintoon ja markkinointiin. Esimerkkinä kaupanalan yrityksissä seurataan tuotteiden kannattavuutta ja asiakkaiden tekemiä ostoksia, kun taas tietoliikenneyhtiöissä voidaan seurata asiakkaiden puhelinkäyttäytymistä ja datan siirtoa. Yleisimmin tietovaraston käyttäjiä ovat johtotehtävissä työskentelevät henkilöt. He kuitenkin harvemmin tuottavat itse tietovarastoja ja raportteja, vaan ratkaisujen on oltava valmiita ja käytöl-

tään käteviä. [2, s. 33–34.] Monesti tietovarastoja tuottamaan palkataan osaavat henkilöt organisaation sisältä tai työ voidaan ulkoistaa BI-palveluihin erikoistuneelle yritykselle. Opinnäytetyön toimeksiantajana toimiva OlapCon Oy on yksi näitä palveluja tarjoavista firmoista.

Tietovarastointiin kuuluu tiedon hakemista operatiivisista tietokannoista, ja mahdollisesti myös tiedostoista, sekä niiden muokkaus haluttuun muotoon, minkä jälkeen tiedot ladataan tietovarastotietokantaan. Tätä prosessia kutsutaan ETL-prosessiksi. Tietovarastoa päivitetään niin, että sinne lisätään joka kerta uutta tietoa ja näin säilytetään historiatiedot, mikä mahdollistaa mm. trendien kuvaamisen. Tietovaraston päivitys tehdään säännöllisin väliajoin, yleensä kerran päivässä. Tietovarastossa olevaa tietoa analysoidaan ja raportoidaan BI-työkaluilla. [1, s. 14–15; 2, s. 29–31.]

Ennen BI-ratkaisujen toteutusta joudutaan usein kuitenkin kuvaamaan tietovaraston tietoa. Tätä tiedon kuvausta kutsutaan metatiedoksi (metadata). Metatieto helpottaa tiedon hyödyntämistä kertomalla mitä kukin tietue oikeasti sisältää. Tietovarastossa voi olla epäselviä tai moniselitteisiä kenttien nimiä ja näihin saadaan selko metatiedon avulla. Tyypillisintä metatietoa ovat tiedon nimi, määritelmä, tietotyyppi, pituus, taulun nimi ja taulun kuvaus. Yleensä metatieto tallennetaan omaan metatietokantaansa. [2, s. 110–111.] Tässä opinnäytetyössä metatietoa mallinnettiin IBM Cognoksen Framework Managerilla, jolla tiedoista rakennettiin framework-paketti, jota käytettiin edelleen BI-työkaluilla toteutusten tekemiseen. Framework Managerista on kirjoitettu lisää luvussa 3.2.

Business Intelligence -työkaluilla saadaan aikaan ratkaisuja, jotka nopeuttavat ja helpottavat yrityksen päätöksentekoa. Ratkaisuna voi olla esimerkiksi valmis raportti tai taulukkolaskentatiedosto. Käytettävyydeltään ja visuaalisuudeltaan hyvin toteutetut ratkaisut eivät vaadi loppukäyttäjältä muuta kuin hiirellä klikkailua ja he saavat eteensä tarvitsemansa tiedot. [1, s. 74.] BI-työkalun lähteenä voivat olla esimerkiksi suorat kyselyt tietovarastoon tai OLAP-kuutio. Kuvassa 1 on esitetty tiivistettynä tietovarastoinnin ja BI:n prosessin arkkitehtuuri.



Kuva 1. Tietovarastointi ja BI -arkkitehtuuri [3.]

Kuvan 1 esimerkissä tietolähteitä on viisi erilaista. Näistä poimitaan tarvittava tieto ja ETL-prosessissa tietoa varmistetaan oikeaksi, puhdistetaan, muokataan, summataan ja lopuksi ladataan tietovarastoon. Tietovarastoina voi olla myös pienempiä datamartteja, kuten kuvassa on esitetty, ja niistä kerrotaan lisää luvussa 2.2. Tietovarastoinnin jälkeen voidaan tuottaa erilaisia BI-ratkaisuja, kuten OLAP-analyysiä, perusraportointia ja dashboardeja.

2.1 SQL ja relaatiotietokannat

SQL-kieli on standardisoitu tietokanta- ja tietovarastokehityksessä käytetty ohjelmointikieli. Ensimmäinen versio SQL-standardista tuli vuonna 1986 (SQL:86) ja uusin versio SQL-standardista on SQL:2011 (ISO/IEC 9075:2011). Vaikka kyselyiden tekeminen tietokantoja vasten onkin SQL:n yleisin käyttöalue, se taipuu ohjelmointikielenä paljon muuhunkin. Sillä voidaan mm. määritellä ja muuttaa tietokannan rakennetta, lisätä, muuttaa ja poistaa tietoja sekä hallita tietokannan käyttöoikeuksia. Isoilla ja pienillä kirjaimilla ei ole merkitystä

SQL:ssä muualla kuin merkkijonojen sisällä. Yleensä kuitenkin on tapana kirjoittaa SQL:n omat komennot isoilla kirjaimilla ja vieläpä omille riveilleen, jotta koodi pysyy lukemisen kannalta selkeämpänä. Esimerkiksi SQL-koodi "SELECT Kenttä FROM Testi" hakee Kenttä-sarakkeen sisällön Testi-nimisestä taulusta. [4.]

Opinnäytetyössä on käytetty Microsoft SQL Serverin omaa versiota SQL-kielestä nimeltään Transact-SQL, tai lyhennettynä T-SQL. Microsoftin omistama T-SQL eroaa avoimesta SQL:stä niin, että siihen on lisätty ominaisuuksia, kuten proseduurillisen ohjelmoinnin elementtejä ja mahdollisuus asettaa paikallisia muuttujia. Nämä ominaisuudet helpottavat varsinkin pidempien skriptien kirjoittamista ja lukemista. Siihen on lisätty myös enemmän funktioita mm. matemaattisiin operaatioihin, merkkijonoihin sekä päivämäärä- ja aikaoperaatioihin. On suositeltavaa käyttää T-SQL:ää Microsoft SQL Serverillä työskennellessä, jotta yhteensopivuudesta voidaan olla varmoja. [5.]

```
SELECT EmployeeID, FirstName, LastName, HireDate, City FROM Employees
WHERE (FirstName NOT LIKE 'M%') AND (FirstName NOT LIKE 'A%')
```

EmployeeID	FirstName	LastName	HireDate	City
1	Nancy	Davolio	1/5/1992 12:00:00 AM	Seattle
3	Janet	Leverling	1/4/1992 12:00:00 AM	Kirkland
5	Steven	Buchanan	17/10/1993 12:00:00 AM	London
7	Robert	King	2/1/1994 12:00:00 AM	London
8	Laura	Callahan	5/3/1994 12:00:00 AM	Seattle

Kuva 2. SQL-kysely ja sen tulos [6.]

Kuvassa 2 on esimerkkikysely ja sen palauttama tulos. Koodilla on haettu Employees-taulusta työntekijäavain (EmployeeID), etunimi (FirstName), sukunimi (LastName), työsuhteen alkamispäivämäärä (HireDate) sekä kotikaupunki (City) sellaisilta työntekijöiltä, joiden etunimi ei ala M tai A-kirjaimella. SELECT-komento on tiedon hakemista varten ja FROM-komento määrittelee, mistä tietokannan taulusta tieto haetaan. Näiden väliin luetellaan sarakkeet, joiden tieto halutaan nähdä. WHERE-komennon jälkeen määritellään mahdolliset suodatukset ja tässä on käytetty LIKE-komentoa, johon määritellään vertauskohteeksi

merkki tai merkkijono ja sitten haetaan sellaiset sarakkeet, joista tämä kohde löytyy. Lisäämällä NOT komennon eteen se toimii toisinpäin, eli hakee kohteet joissa merkkiä tai merkkijonoa ei esiinny. SQL sisältää todella paljon erilaisia komentoja ja kuvassa 2 on vain muutama niistä. [6.]

Kuvassa 2 näkyvä tulos on myös oiva esimerkki relaatiotietokannan taulusta ja sen sisällöstä. Relaatiotietokannoissa on useita vastaavanlaisia tauluja, jotka sisältävät useita rivejä tietoa jaoteltuna eri sarakkeisiin, joista käytetään myös nimitystä kenttä. Ensimmäisessä sarakkeessa on yksilöivä tieto eli avain (tai lyhyemmin id) ja muissa sarakkeissa avaimeen liittyvät tiedot. Avainten avulla relaatiotietokannan taulut voidaan linkittää toisiinsa, sillä yhden taulun pääavain (Primary key) löytyy yleensä jostain muusta taulusta viiteavaimena (Foreign key). Linkityksen avulla voidaan tehdä monipuolisia kyselyitä joissa tietoa on yhdistelty useasta taulusta. Näitä relaatiotietokannan ominaisuuksia käytetään hyödyksi tietovarastoinnin ETL-vaiheessa. On olemassa monia muitakin tietokantamuotoja, mutta relaatiotietokanta on kaikista tietokantamuodoista yleisin. [6.]

Perusjärjestelmien (esim. ERP tai CRM) operatiivisina tietokantoina on siis lähes aina SQL-pohjainen relaatiotietokanta. Operatiivisiin tietokantoihin tietoa tulee päätteiden ja työasemien kautta tietoa syöttämällä, kun taas tietovarastoihin tieto menee ajastettujen latausten eli eräajojen kautta. Tietovarastot toteutetaan lähes aina relaatiotietokantatekniikoilla, mutta tietovarastot ovat silti hyvin erilaisia operatiivisiin kantoihin verrattuna. Jos tietovarasto toteutetaan ainoastaan täsmälleen relaatiokannan malliseksi, siitä jää pois mm. moniulotteisen analyysin tuki. Yleinen tietovaraston malli onkin tähtimalli, joka mahdollistaa moniulotteisen analyysin säilyttämällä silti relaatiokannan taulurakenteen, eli mm. aiemmin mainitut avaimet. OLAP-kuutiot ovat varsinaisia moniulotteiseen analyysiin tarkoitettuja erikoiskantoja. [2, s. 45, 56–57.] Tähtimallista ja OLAPista on kerrottu lisää myöhemmissä luvuissa.

2.2 ETL-prosessi ja tietovarastokehityksen tekniikat

Tietovarasto eli Data Warehouse on yhdistelmä monien eri operatiivisten järjestelmien tiedoista ja se on yleensä useiden käyttäjien yhteisessä käytössä. Tietovaraston kehitys aloitetaan suunnittelulla, jota seuraa ETL-prosessin toteutus. Kuten aiemmin mainittiin, tietoja voidaan tarvita useammasta eri lähteestä, joissa kaikissa tieto on eri tavalla säilöttynä. Aivan ensimmäisenä on suunniteltava, mistä tietolähteistä mitäkin tietoa haetaan, millä tavoin sitä on muokattava ja minne se on ladattava tietovarastossa. Esimerkiksi aikaleimoista voidaan laskea ajan pituuksia vaikkapa päivän tai tunnin tarkkuudella (T-SQL:n DATEDIFF() funktio) omaan sarakkeeseensa. Monia tietoja tuodaan kuitenkin tietovarastoon myös sellaisenaan ilman minkäänlaista muokkausta. [2, s. 78–79.]

ETL-vaiheen toteutukseen käytetään yleensä siihen tarkoitettuja ETL-työkaluja, kuten Microsoft SQL Serverin SSIS:ää (luku 3.1), jotka helpottavat latauksien tekoa graafisen käyttöliittymänsä ansiosta. ETL-lataukset on myös mahdollista toteuttaa ohjelmoimalla ja tekemällä tietokantaan talletettuja proseduureja (Stored procedure). Nämä proseduurit voidaan ajastaa ajettavaksi sopivana ajankohtana, mutta yleensä öisin, jotta päiväsaikaan tietovarasto on käyttövalmiina raportointiin. Microsoftin SQL Serverissä proseduureja ohjelmoidaan aiemmin mainitulla T-SQL:llä. Ohjelmoinnin haittapuolena on se, että se on työläämpää varsinkin jos tietovarastosta on tarkoitus tulla iso ja tietoa muokataan paljon. Yhden henkilön tekemät SQL-koodit ovat myös joskus muille työntekijöille epäselviä lukea ja näin jatkokehityksestä tulee vaikeampaa. Onkin hyvä tapa kirjoittaa koodiin kommentteja selkeyttämään, mitä missäkin kohdassa tapahtuu. ETL-työkalut taas muodostavat itsestään koodia samalla kun käyttäjä muodostaa latausta graafisessa käyttöliittymässä, joten alla piilevä koodi on aina tyylillään samanlaista ja näin ollen henkilöriippuvuus vähenee. [2, s. 76–78; 1, s. 53–54.]

ETL-prosessin ensimmäinen vaihe on tiedon poiminta ja mahdollisesti siirto, jos tietovarasto tulee eri palvelimelle kuin tietokannat. Yleensä poiminta on suoraviivaista lukemista tietokannan tauluista (SELECT-komento). Poiminnan jälkeen

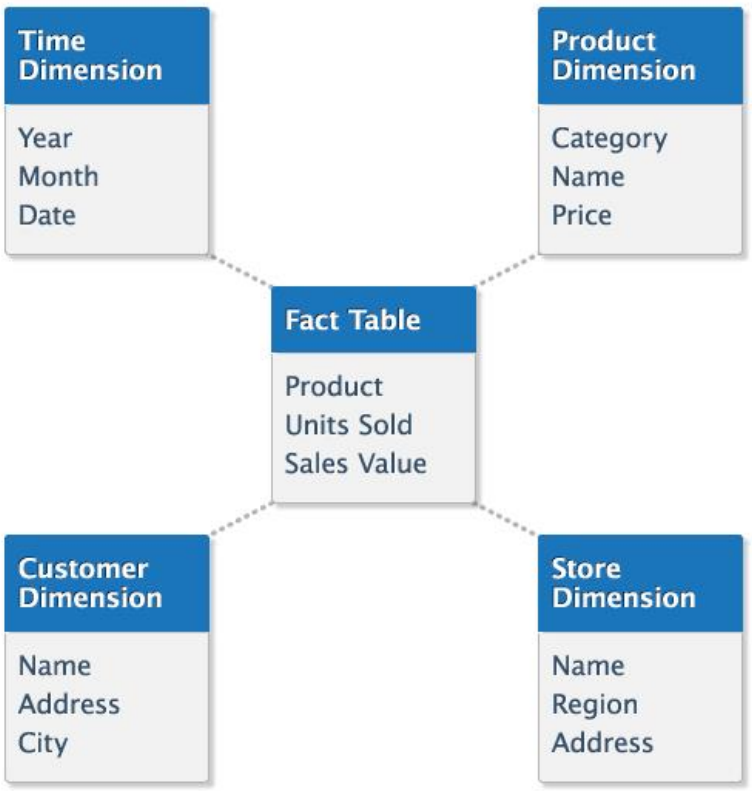
tulee ETL:n työläin vaihe eli muokkaus. Muokkausvaiheessa suoritetaan monta eri tehtävää, joista ensimmäisenä tarkastus, että poimittu tieto on varmasti oikeaa. Tarkistetaan myös, että kysely ei palauta NULL-arvoja, eli tyhjiä kenttiä. Joissain tapauksissa voi olla tietysti mahdollista, että sallitaan NULL-arvot kenttiin, joiden tieto ei ole pakollista tai tiedetään, että kaikille riveille ei tätä tietoa löydy. Muita tarkistuksia ovat mm. tietojen muotojen tarkistus ja tuplarivien poisto. Virheelliset rivit voidaan jättää kokonaan pois, jos itse kysely todetaan virheettömäksi. Toinen vaihtoehto on merkitä virheelliset rivit jollain erottimella ja säilyttää ne. [1, s. 55–56.]

Kun tieto on tarkistettu, voidaan aloittaa sen varsinainen muokkaus. Monet tietovarastoon menevät tiedot kelpaavat ihan sellaisenaan, mutta joitakin tietoja joudutaan yleensä muuntamaan. Muuntaminen voi olla tietojen yhdistämistä useammasta taulusta yhteen tauluun tietovarastossa. Esimerkiksi asiakastietoja voi olla levällään useamman järjestelmän tietokannoissa ja nämä voidaan koota samaan tauluun, kun vain tiedossa ovat hakuperusteet, joilla yhden asiakkaan tiedot saadaan haettua kaikista tietokannoista. Myös avainsarakkeiden ja muiden sarakkeiden nimien yhdenmukaistaminen tapahtuu tässä vaiheessa. Eri järjestelmissä voi olla samaa asiaa tarkoittava avainsarake nimettynä eri tavalla tai itse avaimen arvo voi olla eri, joten toimivuuden vuoksi nämä tiedot on hyvä yhdenmukaistaa tietovaraston jokaisessa taulussa. Joistain tiedoista voidaan myös joutua ottamaan vain osa, esimerkiksi henkilötunnuksesta pelkkä syntymäaika ja päinvastoin voi olla tarve tehdä lisäyksiä joihinkin tietoihin. Tietovaraston tauluihin halutaan myös yleisesti valmiiksi laskettuja tunnuslukuja ja mittareita sekä summatauluja. Valmiiksi voidaan laskea esim. asiakaskate (liikevaihto-kustannukset) ja vaikkapa myynnin lukuja voidaan summata kuukausitasolle omaan summatauluunsa. [1, s. 56–57]

Muokkausvaiheen jälkeen seuraa tietojen lataus (INSERT-komento) tietovarastoon tauluihin. Staattisten taulujen ja dimensiotaulujen rivit päivitetään yleensä tekemällä lataus edellisten tietojen päälle. Tapahtuma- tai faktatyyppiset taulut taas päivitetään yleensä niin, että vain uudet rivit lisätään edellisten tietojen perään. Näin säilytetään historiatietoja. Dimensio- ja faktataulut liittyvät tähtimallisiin tietovarastoihin ja niistä kerrotaan lisää myöhemmin tässä luvussa. Kun

lataus on tehty ja todettu toimivaksi, se ajastetaan ajettavaksi haluttuina aikoina. Ajastukset voidaan tehdä esim. ETL-työkalujen tai käyttöjärjestelmän ajastimilla. Microsoft SQL Serverin SSMS:llä voidaan ajastaa ohjelmoimalla tehtyjä proseduureja tai SSIS:llä tehtyjä ETL-paketteja. Kun latauksen ajastus on tehty ja ETL-prosessi saatettu päätökseensä, tietovarasto on valmis ja sitä voidaan hyödyntää BI-käytössä. [1, s. 58.]

Varsinaisen tietovaraston lisäksi myös pienemmät datamartit (Data Marts kuvassa 1) ovat tietovarastoja. Datamartit ovat useimmiten aiheeltaan rajattuja ja voivat keskittyä esimerkiksi pelkästään talouden raportointiin. Tästä syystä datamartit on suunniteltu vain tietyn käyttäjäryhmän hyödynnettäväksi BI-tarkoituksessa. Datamartit voivat olla summa- tai tapahtumatasoisia ja ne toteutetaan yleensä relaatiotietokantatuotteilla esim. tähtimallin (kuva 3) mukaisesti. Tähtimallin datamarteista voidaan myös kätevästi tehdä kuutioratkaisuja OLAP-käyttöä varten. [1, s. 23–24.]



Kuva 3. Tähtimalli, jossa on yksi faktataulu ja neljä dimensiota [7.]

Tähtimallissa tieto on kerätty faktatauluun ja siihen liittyviin dimensiotauluihin. Tätä sanotaan moniulotteiseksi malliksi. Faktana on jokin mitattava tieto, kuten myytyjen tuotteiden määrä ja hinta. Faktataulun rivit sisältävät faktan lisäksi avaimia, jotka linkittyvät dimensiotauluihin ja faktataulut ovat yleensä todella suuria verrattuna dimensiotauluihin. Dimensiotaulut taas sisältävät vastaavat avaimet ja niiden selitykset. Esimerkiksi kuvassa 3 esitetyn mallin faktataulun rivit sisältävät päivämäärän, joka toimii aika-avaimena, sekä tuoteavaimen, asiakasavaimen ja myymälä-avaimen. Rivien faktana on myytyjen tuotteiden kappalemäärä sekä hinta. Dimensiotaulujen ja faktataulun välillä on yksi-moneen-suhde, eli dimensiotauluissa sama avain esiintyy vain yhden kerran ja faktataulussa mahdollisesti todella monellakin rivillä. Dimensiotauluissa avaimet (kutsutaan myös id:ksi) ovat siis aina yksilöityjä, jotta yhden avaimen takaa ei löydy montaa eri tietoa. Asiakasdimensiossa avaimen takaa löytyy asiakkaan nimi, osoite ja kaupunki. Myymälädimensio sisältää myymälän nimen, alueen ja osoitteen. Tuotedimensiossa on tuotteen kategoria, nimi ja hinta. Aikadimensiossa on tarkat päivämäärät ja eriteltynä niiden vuosi, kuukausi sekä mahdollisesti kvartaali. [8.]

Kun tähtimalliin tehdään kyselyitä, fakta- ja dimensiotaulun avaimet ”joinataan” eli liitetään keskenään niin, että F-taulun AsiakasID vastaa asiakastaulun AsiakasID:tä. Näin voidaan tehdä vaikkapa kysely, jossa haetaan kaikkien tietyssä kaupungissa asuvien asiakkaiden ostosten hinnat summana. Tähtimallilla toteutettu tietovarasto on nopeampi käyttää, koska kyselyt pysyvät yksinkertaisina. Se on myös suorituskyvyllisesti kevyt vaihtoehto, koska faktataulun ja dimensiotaulujen välillä on vain yksi liitos. Tähtimalli onkin yksinkertaisin tietovaraston toteutusmalli. Toinen yleinen vaihtoehto olisi tehdä lumihiutalemalli, mutta se on hieman monimutkaisempi ja sitä ei ole hyödynnetty tässä opinnäytetyössä. [9, s. 45–48.]

2.3 Business Intelligence ja OLAP

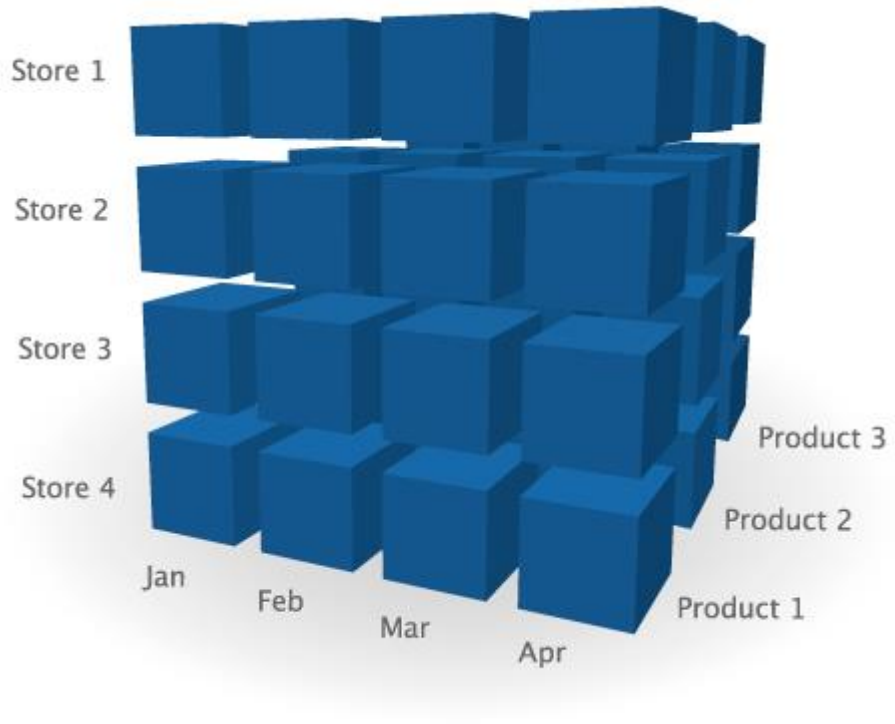
BI:n avulla yritysten ja organisaatioiden henkilöstö pystyy tarkastelemaan liiketoimintaa kuvaavaa informaatiota ja tekemään sen perusteella varmempia pää-

töksiä, jotka ohjaavat yrityksen liiketoimintaa oikeaan suuntaan. Tämän lisäksi muita BI:n tavoitteita ovat tiedon nopea saatavuus eri raportointityökalujen avulla sekä organisaation strategian ja tavoitteiden saavuttamisen tukeminen. BI-ratkaisut myös lisäävät käyttäjien omatoimisuutta tiedon hankinnan suhteen, kun kaikki raportointityökalut ovat käyttövalmiina koko henkilöstölle. Käyttäjät voivat itse hakea tietoa niistä tietolähteistä, joihin heillä on pääsy ja tehdä raportteja, joita he pystyvät tarvittaessa jakamaan edelleen muille yrityksen työntekijöille. Myös tehokkuus parantuu ja kustannukset vähentyvät, koska raportit voidaan ajastaa päivittymään ja niiden saaminen esim. sähköpostitse voidaan automatisoida. Tämä automatisointi säästää paljon rahaa ja aikaa, kun manuaalista raporttien laatimista ei tarvita. [1, s. 80–81.]

Raportointi on tärkein tiedon hyödyntämismuoto BI-ratkaisuissa. Nykypäivänä raportointia tehdään enimmäkseen selainpohjaisissa ympäristöissä (esim. IBM Cognos-ympäristö), joissa raportit toteutetaan ja julkaistaan. Uusien raporttien teko aloitetaan tekemällä tietovarastoon kysely, joka on SQL-kielellä tehty relaatiotietokantahaku. Nykyään raportointityökalut on käyttöliittymältään suunniteltu niin, ettei käyttäjän tarvitse itse kirjoittaa kyselyä, vaan kaikki tapahtuu graafisessa käyttöliittymässä. Hyvän raportointityökalun edellytyksenä onkin, että sen käyttö on helppoa ja vaivatonta. Yleisimmin raportit sisältävät taulukkomuodossa esitettyä tietoa ja sitä kuvaavia graafeja, esim. pylväsdiagrammeja. Jos tietoa halutaan analysoida monelta kantilta ja nopeasti, voidaan käyttää tässäkin opinnäytetyössä hyödynnettyjä OLAP-työkaluja moniulotteisen analyysin tuottamiseen. [1, s. 86–87.]

OLAP, eli Online Analytical Processing, tarkoittaa suomeksi moniulotteista analysointia. [1. s. 91.] OLAP-työkaluilla tehdään kyselyitä, analysoidaan ja raportoidaan tietoa. OLAP-työkalujen avulla käyttäjät voivat tehdä moniulotteista analyysia tiedoista, eli kerralla tarkkailussa on useampi kuin yksi dimensio. Voidaan esimerkiksi tutkia, kuinka monta kappaletta jotakin tuotetta on myyty tietyssä toimipisteessä tiettynä aikana. Tällöin dimensioina ovat aika, tuote ja myymälä. Kuvan 4 esimerkkikuutiossa on käytetty juuri näitä ulottuvuuksia. Kuutiot ovat OLAP-järjestelmien keskeinen osa. Niissä summataan valmiiksi faktat jokaisilta dimensioiden tasoilta, joita tähtimallista saadaan. Tämän kuutiorakenteen avulla

OLAP-järjestelmät antavat nopeasti vastauksia esitettyihin kyselyihin, koska vastaukset ovat käytännössä jo valmiina odottamassa. [7.]



Kuva 4. OLAP-kuutio [7.]

Kuution sisältöä voidaan lähteä tutkimaan vaiheittain. Valitaan aluksi vaikkapa pelkästään yksi summattu luku (esimerkiksi myytyjen tuotteiden määrä) ilman yhtään dimensiota. Tällöin kysely palauttaa pelkästään summan kaikista myydyistä tuotteista. Jos kyselyyn lisätään dimensioksi aika, se palauttaa myydyt tuotteet esim. vuosittain. Aikaa voidaan myös tarkastella kvartaaleittain, kuukausittain, viikoittain jne. riippuen siitä, mikä dimension hierarkiataso on valittu käyttöön. Lisätään vielä kaksi dimensiota: tuote ja myymälä. Nyt kuution kysely palauttaa aiemmassa kappaleessa mainitun tiedon myydyistä tuotteista per myymälä tiettyinä ajanjaksoina. Kuvassa 4 esitetystä kuutiosta voitaisiin hakea, että kuinka monta kappaletta tuotetta 1 (Product 1) on myyty myymälässä 2 (Store 2) helmikuussa (Feb). [1. s. 91–93.]

Kuutiot eivät välttämättä kuitenkaan rajoitu kolmeen dimensioon, vaan niitä voi olla huomattavasti enemmän. Yleisesti suositaan 3-8 dimensiota, jotta kuutio

pysyisi hallittavuuden ja käytettävyyden kannalta selkeänä. Porautuminen ja hierarkiat ovat termejä, jotka liittyvät kuutioihin ja OLAP-ratkaisuihin. Kuutioita rakennettaessa dimensioihin määritellään tasoja, jotka muodostavat hierarkian. Tuotedimensiossa hierarkian voivat muodostaa esimerkiksi tuoteryhmä, tuoteperhe ja itse tuotteet. Käyttäjät voivat porautua hierarkiassa tasolta toiselle, eli yhden tuotteen tarkastelusta voidaan porautua ylöspäin katsomaan sen tuotteen tuoteperheen kaikkien tuotteiden myyntiä ja vielä edelleen tuoteperheiden kokonaismyyntiä. Samalla tavalla voidaan porautua alemmalle tasolle tarkastelemaan valitun tuoteperheen tuotteiden myyntiä. [1. s. 93.] Esimerkkinä vielä kuvan 3 tähtimallin myymälädimensiotaulusta saataisiin kuutiossa myymälädimensiolle pieni kahden tason hierkarkia, jossa on alueet ja niiden alla myymälät.

Revenue		Central Europe	Asia Pacific	Americas	Northern Europe	Southern Europe
2004	Department Store	1,215,511	243,980	4,298,900	534,445	652,363
	Direct Marketing	526,424	198,834	198,379	232,057	0
	Equipment Rental Store	82,199	0	220,663	42,724	0
	Eyewear Store	60,768	1,461	73,113	0	0
	Golf Shop	1,101,302	803,953	498,779	276,717	205,812
	Rank (Golf Shop)	1	2	3	4	5
	Outdoors Shop	4,895,268	149,696	5,611,253	1,639,240	2,269,355
	Sports Store	1,231,342	611,248	2,909,059	612,184	654,067
	Warehouse Store	1,093,757	0	1,601,120	0	0

Kuva 5. Yksinkertainen OLAP-raporttinäkymä [10.]

Kuvassa 5 näkyy esimerkki yksinkertaisesta OLAP-raporttinäkymästä. Siihen on tuotu kuutiosta mittariluvuksi liikevaihto ja dimensioina ovat aika, kaupat ja alueet. Kuvassa on käytetty ristiintaulukointia (crosstab), johon mittariluku eli liikevaihto on tuotu keskelle ja riveille on tuotu aika- ja kauppadimensiot. Aluedimensio on tuotu sarakkeisiin. Jos kuutioon olisi määritelty dimensioille hierarkiat, olisi porautuminen raportilla mahdollista. Tällöin voitaisiin esimerkiksi Pohjois-Euroopan kohdalta klikkaamalla porautua alemmalle tasolle tarkastelemaan siihen alueeseen kuuluvien maiden tilannetta.

3 Toteutuksessa käytetyt teknologiat

Opinnäytetyö tehtiin enimmäkseen OlapConin tiloissa ja työkoneena oli yrityksen kannettava tietokone. Välillä työtä toteutettiin VPN-yhteyden kautta etätyöskentelynä. Kaikki opinnäytetyössä käytetyt ohjelmistot olivat OlapConilla valmiina ja asennettu niin, että käytössä olleet Microsoft SQL Server 2012 ja IBM Cognoksen BI-tuotantojärjestelmät olivat eri palvelimilla. Opinnäytetyön tekemiseksi annettiin tarvittavat käyttöoikeudet järjestelmiin. Cognoksen ohjelmistot sitoutuvat yhteen IBM Cognos Connectionissa, joka on Cognoksen pääportaali. Portaalia käytetään nettiselaimella ja sieltä hallinnoidaan Cognoksen eri komponentteja ja sinne julkaistaan raportoinnin tietolähteitä sekä itse raportteja. Tässä luvussa on kerrottu yleisesti perusperiaatteet jokaisesta opinnäytetyössä käytetystä ohjelmistosta siinä järjestyksessä, kuin niitä käytettiin.

3.1 Microsoft SQL Server 2012

Microsoftin SQL Server on tietokantojen ja samalla tietovarastojen hallintaohjelmisto. SQL Serveristä tulee muutaman vuoden välein uusi julkaisu ja siitäkin on aina olemassa monta eri versiota. Nyt käytössä oli SQL Server 2012:n Business Intelligence Edition. Suurimpana erona perus-, eli Standard-versioon on se, että siitä löytyy enemmän Microsoftin BI-työkaluja, kuten PowerPivot for Sharepoint ja PowerView [11]. SQL Server sisältää SSMS-tietokantahallinnan lisäksi paljon muitakin teknologioita, kuten SSIS ETL -työkalun, SSAS OLAP -työkalun mm. kuutioiden tekoon ja SSRS-raportointityökaluja [12]. Vaikka SQL Server sisältääkin Microsoftin tarjoamia BI-teknologioita, niin tässä opinnäytetyössä käytettiin ainoastaan sen tietokantahallintaa (SSMS) ja ETL-työkalua (SSIS) tietovaraston toteutukseen ja BI-puoli toteutettiin toimeksiantajan valitsemilla IBM:n Cognos-tuotteilla.

SSMS:llä voidaan luoda, tutkia, muokata ja ylläpitää tietokantoja sekä tietovarastoja. Siinä voidaan tehdä ohjelmoimalla SQL-kieltä käyttäen kyselyjä tieto-

kantoihin. SSMS toimii SQL Serverin pääkäyttöliittymänä ja sillä pystytään hallitsemaan kaikkia SQL Serverin komponentteja. SSIS:llä tehdään projekteja, joissa toteutetaan ETL-prosessi ja luodaan sen pohjalta SSIS-paketteja. Yksi paketti voi sisältää useita latauksia monista eri tietokannoista ja niissä on aina sekä lähdekanta että kohdekanta. Latauksessa voi olla todella monta muokkausvaihetta, joissa dataa yhdistellään ja muutetaan tarvittavaan muotoon. SSIS-paketteja voidaan ajastaa SSMS:n puolella ajettavaksi tiettynä aikana ja näin saadaan tietovarasto päivitettyä tuoreella tiedolla. SSIS:n käyttöliittymä on ETL-työkalulle tyypillisesti graafinen ja käyttäjäystävällinen, joten ETL-prosessin tekeminen on sillä helppoa.

Opinnäytetyön toteutuksessa käytettiin kuitenkin suurimmaksi osaksi SSMS:ää myös ETL-prosessin toteutuksessa, eli poiminta, muokkaukset ja lataukset suoritettiin ohjelmoimalla T-SQL kyselyitä ja tallentamalla niistä sitten proseduureja tietokantaan. Nämä proseduurit ajastettiin SSMS:llä. Yksi paketti tehtiin myös SSIS:llä, eli molempia ETL-prosessin toteutustapoja, ohjelmointia ja valmista työkalua, hyödynnettiin.

3.2 IBM Cognos Framework Manager

Framework Manager on metatiedon mallinnustyökalu. Sillä luodaan ns. framework-paketti, joka sisältää metatiedon mahdollisesti monestakin eri tietolähteestä. Metatietomallissa tehdään tarvittaessa muutoksia taulujen ja kenttien nimiin, tai voidaan luoda myös kokonaan uusia laskettuja kenttiä tai lisätietokenttiä. Kun malli on valmis, valitaan, mitä tietoa tarvitaan ja halutaan sisällyttää pakettiin. Tämä luotu paketti julkaistaan Cognoksen Connection-portaaliin ja sitä voidaan käyttää BI-työkalujen tietolähteenä. Kun paketissa on kenttien ja taulujen nimet muokattu käyttäjän kannalta selkeiksi, saadaan raportoinnin käyttöön soveltuva tietolähde, vaikka tietovarastossa itsessään olisi epäselviä kenttänimiä ja taulujen nimiä. [13.]

Kaikki raportoinnissa tehtävät kyselyt menevät tässä tapauksessa framework-paketin kautta, eikä loppukäyttäjä ole missään vaiheessa suoraan tekemisissä

tietovaraston kanssa. Perusraportoinnin tarpeisiin Framework Managerilla voidaan kätevästi luoda myös suhteet eri taulujen välille käyttämällä taulujen avainkenttiä. [13.] Jos tarkoituksena on tehdä paketin pohjalta kuutio Transformerilla, niin pakettiin ei tarvitse tehdä taulujen välisiä suhteita, sillä kuutiota muodostaessa niillä ei ole merkitystä, vaan kuution dimensiot ja faktat määritellään eri tavalla Transformerin puolella.

3.3 IBM Cognos Transformer

Transformer on OLAP-tiedon mallinnukseen tarkoitettu työkalu. Transformerilla luotavasta mallista tehdään kuutioita, joita Cognoksen tapauksessa kutsutaan PowerCubeiksi. Kuutiota tehdessä on valittava tietolähde, eli tämän opinnäytetyön tapauksessa Cognoksen portaaliin julkaistu framework-paketti, josta kyselyitä tekemällä haetaan tarvittavat tiedot Transformeriin. Transformerin avulla voidaan luoda malliin automaattisesti dimensiot ja mittariluvut, sillä Transformer tunnistaa tuoduista tauluista dimensioiden avaimet ja faktataulujen luvut. Tämä on myös toki mahdollista tehdä manuaalisesti, jolloin saa varmasti tarkoituksenmukaisen tuloksen. [14.]

Kuution mittariluvut on mahdollista allokoida automaattisesti kuution dimensioihin dimensiotaulujen ja faktataulujen avainkenttien avulla. Jos halutaan hyödyntää tätä Transformerin automatiikkaa, niin faktatauluissa täytyy olla dimensioiden avainsarakkeet samoilla nimillä kuin itse dimensiotauluissa. Kuten aikaisemmin luvussa 2.2 mainittiin, onkin hyvä pyrkiä jo ETL-prosessia tehdessä nimeämään taulujen sarakkeet samalla nimellä, jos ne sisältävät saman tiedon. Kun kaikki mittariluvut on allokoitu jokaiselle dimensiolle, voidaan mallista rakentaa kuutio. Kun kuutio tehdään, se tallentuu oletushakemistoon tiedostona, jota voidaan siirtää tarvittaessa ympäristöstä toiseen kuin mitä tahansa tiedostoa. Nämä kuutiot julkaistaan tietolähteeksi IBM Cognoksen Connection-portaaliin raportointityökalujen käyttöön OLAP-raportointia ja moniulotteista analysointia varten. [14.]

3.4 IBM Cognos Analysis Studio

Analysis Studio on yksi IBM Cognoksen tarjoamista raportointityökaluista ja se on tarkoitettu nimenomaan moniulotteiseen analyysiin. Analysis Studio on selainpohjainen sovellus, jota käytetään IBM Cognos Connection -portaalin kautta. Käyttöliittymä on hyvin interaktiivinen ja analyysejä voidaan tehdä ja muokata vetämällä hiirellä haluttuja mittarilukuja ja dimensioita tietolähteestä raportille esim. ristiintaulukkoon. Tiedon käsittely ja raporttien luominen on tästä syystä todella yksinkertaista ja helppoa sekä nopeaa, kun tietolähteenä on moniulotteinen malli, OLAP-kuutio. [15.]

Analysis Studiolla voidaan kätevästi mm. vertailla summa- ja rivitason tietoja sekä toteumia ja budjetteja. Raportille voidaan myös muodostaa laskettuja arvoja, esim. myynnin kasvua verrattuna edelliseen kuukauteen ja porautuminen on mahdollista klikkaamalla taulukoita niistä dimensioista, joihin halutaan porautua. Raportille tulevat tiedot saadaan myös rajattua asettamalla suodattimia (filters). Suodattimilla voidaan vaikkapa rajata käyttöön pelkästään vuoden 2014 tiedot. Raporttien tallentaminen ja jakaminen muiden henkilöiden käyttöön onnistuu Cognos-portaalissa, jossa muut työntekijät voivat tarvittaessa itsekin muutella raportilla näkyviä tietoja mieleisikseen ja tallentaa muokkaamansa näkymän itselleen talteen. [15.]

4 Opinnäytetyön lähtöasetelma ja tavoitteet

Jatkokehityksen kohteena oli OlapConin sisäinen talousraportointi. Yrityksessä oli jo kehitetty toiminnanohjausjärjestelmän ja kirjanpidon tietokantoihin pohjautuvaa OLAP-kuutiota, joka oli vielä keskeneräinen ja sen avulla saatiin tarkasteltua yrityksen liikevaihtoa, kustannuksia ja katetta lähinnä aika- ja tilidimensiota vasten. Työssä oli tarkoitus tehdä vastaavanlainen kuutio, mutta siihen täytyi saada lisää dimensioita ja mittarilukuja. Entinen kuutio oli tehty IBM Cognos Transformerin vanhemmalla 7-versiolla ja uusi tultaisiin tekemään 10-versiolla.

7-versiolla tehdyn kuution kääntäminen 10-version kuutioksi olisi ollut mahdollista, mutta päätettiin, että tehdään uusi kuutio alusta alkaen, sillä muutoksia tulee kuitenkin todella paljon. Entinen kuutio ei myöskään ollut täysin valmis ja parissa sen kyselyistä oli vielä virheitä, joiden takia dataa saattoi puuttua ja luvut olivat virheellisiä. Työn päätavoitteena oli asiakasdimension liittäminen liikevaihtoon ja kustannuksiin. Muita tarvittavia dimensioita olivat henkilö ja kustannuspaikka. Työn aikana oli myös tarkoitus selvittää, millä menetelmällä luvut saataisiin jaettua eri dimensioille niin, että ne pysyisivät myös luotettavina.

Vanhan kuution kyselyt oli tehty suoraan tietokantoja vasten ilman tietovarastoa ja siksi sen käyttö oli rajoittunutta. Siitä pystyttiin laskemaan vain yleisesti kate (liikevaihto – kustannukset) halutulle ajalle ja tilille. Tähän haluttiin muutoksia niin, että kate olisi tarkasteltavissa eri dimensioihin allokoituna, esimerkkinä kate per asiakas valitulla aikavälillä. Uuteen OLAP-kuutioon tarvittiin dimensioiksi aika, asiakas, henkilö, tili ja kustannuspaikka. Tarvittavia mittarilukuja olivat liikevaihto, myyntikate, myyntikate-%, käyttökate, käyttökate-%, liikevoitto, liikevoitto-%, kustannukset, myyntikustannus ja käyttökustannus. Kaikki nämä mittarit oli saatava toimimaan jokaista dimensiota vasten.

Kuution toteutukseen tarvittavan tiedon saamiseksi oli ensimmäisenä toteutettava tietovarasto, johon kerättiin dataa sekä toiminnanohjauksen että kirjanpidon tietokantojen tauluista. Tietovarastossa tätä dataa yhdistelemällä ja muokkaamalla saatiin haluttu asiakaskohtainen allokaatio kustannuksille sekä liikevaihdolle ja tätä kautta oli mahdollista laskea kate eri asiakkaille. Aikatauluna oli, että työ valmistuisi kevään 2014 aikana.

5 Työn toteutus

Työn aloitusvaiheessa pidettiin palaveri, jossa käytiin vielä kerran läpi opinnäytetyön tavoite ja lähtökohdat. Suunniteltiin, minkälaista tietovarastoa tultaisiin kehittämään ja käytiin läpi jo olemassa olevaa vanhaa kuutiota. Ensin toteutet-

tiin tietovarastointiprosessi, jonka jälkeen tehtiin paketti Cognoksen Framework Managerilla, josta edelleen kuutio Cognoksen Transformerilla. Kuution pohjalta saatiin kehitettyä raportointia moniulotteisena analyysina. Toimeksiantajan vaatimuksesta työn ETL-prosessissa käytettyjä SQL-skriptejä ei julkaista tässä dokumentissa, vaan on käyty läpi periaatteita, joilla taulut muodostettiin ja kuinka liikevaihto ja kustannukset saatiin allokoitua eri dimensioille. Myöskään yrityksen liiketoimintaan liittyviä lukuja ei esitetä.

5.1 Tietovaraston suunnittelu

Tietovaraston kehityksen alkuvaiheeseen kuului suunnittelua ja määrittelyä tarvittavista tauluista, aivan kuten luvun 2.2 alkupuolella mainittiin. Koska tietovaraston pohjalta tehdään kuutio, tarvittiin sekä faktatauluja että dimensiotauluja. Faktatauluihin haettiin mittareiden lukuarvoja ja dimensioiden id-sarakkeita. Faktataulut voivat kasvaa hyvinkin massiivisiksi (useita miljoonia rivejä), jos dataa on paljon käsiteltävänä, mutta tässä toteutuksessa taulut pysyivät pienempinä. Dimensiotaulut ovat pienempiä ja sisältävät id:n takana olevia lisätietoja, jotka kertovat enemmän ja yksityiskohtaisemmin kyseisestä dimensiosta. Esimerkiksi asiakasdimensiotaulussa oli id:n takana mm. asiakkaan nimi. Tietovarasto toteutettiin luvussa 2.2 kuvatun tähtimallin mukaisesti ja siitä tuli data mart -tyyppinen, sillä se keskittyi yhteen osa-alueeseen (talouden raportointiin) ja sen käyttäjänä tulivat olemaan ainoastaan yrityksen johto.

Tietovaraston taulujen nimeämisessä käytettiin yleistä menetelmää, missä faktataulujen nimen eteen tulee aina ”F” (esim. F_Kustannukset) ja dimensiotaulun nimen eteen ”D” (esim. D_Henkilo). Näin tietovarasto pysyy selkeämpänä. Tietovarastoinnin ETL-prosessi päätettiin toteuttaa proseduureilla, eli ohjelmoimalla T-SQL-kyselyitä suoraan tietovarastoon SSMS:n käyttöliittymää hyödyntäen. Kuten luvussa 2.1 mainittiin, T-SQL mahdollistaa proseduurillisen ohjelmoinnin ja SQL Server käyttää sitä. Tällä tavoin tuli tutustuttua samalla aikaisempaa syvemmin SQL-kielellä ohjelmointiin ja opittua paljon uutta T-SQL:llä ohjelmoinnista. ETL-prosessin loppupuolella tarvittiin kuitenkin vielä lisätä tietovarastoon

tietoa myös yhdestä Excel-tiedostosta ja sen tuomiseen käytettiin Microsoftin ETL-työkalua SSIS:ää.

Vanhaan kuutioon kirjanpidon tietokannasta haetut kustannukset allokoituivat aika-, ja tilidimensioille. Joiltain tileiltä oli mahdollista liittää kustannuksia myös henkilöittäin, mutta asiakkaisiin ja kustannuspaikkoihin ei ollut suoraa liitosta. Liikevaihto taas allokoitui aika-, tili-, ja asiakasdimensioon, mutta sitä ei saatu suoraan liitettyä henkilöihin ja kustannuspaikkoihin. Tämän vuoksi täytyikin miettiä toimeksiantajan kanssa, kuinka nämä faktatiedot saataisiin jaettua dimensioiden kesken. Tulimme siihen tulokseen, että kehitetään prosentteja sisältäviä summatauluja, joiden avulla saadaan allokoitua kustannuksista ja liikevaihdosta osuuksia eri dimensioille. Tietovaraston summatauluista tehtäisiin kuukausitasoisia, koska ei ollut tarvetta tarkastella tietoja viikkojen tai päivien tarkkuudella.

Keskeisenä asiana ETL-prosessissa oli erilaisten jakoperusteiden, eli tässä tapauksessa prosentteja sisältävien summataulujen luonti. Summataulussa oli esimerkiksi kuukausittainen prosentti, kuinka paljon työtä henkilöltä on mennyt millekin asiakkaalle. Tällä prosentilla voitiin sitten faktatauluja muodostettaessa jakaa kustannuksia kullekin asiakkaalle ja henkilölle. Jokaisen tietovarastoon tarvittavan taulun muodostamiseen käytetyt T-SQL-skriptit säilytettiin ja tallennettiin tietovarastoon proseduureiksi. Proseduurit pystytään ajastamaan SSMS:llä tietovaraston päivittämiseksi. Tietovarastoon päätettiin tuoda tietoja vuoden 2013 alusta alkaen, sillä sitä aikaisemmat tiedot olivat hieman eri tavalla tallennettuja ja puutteellisia, joten joitain nykytilanteen kannalta oleellisia tietoja saattaisi puuttua.

5.2 ETL-prosessi

Tässä luvussa on kerrottu ETL-prosessin eteneminen ja periaatteet, joilla tietovaraston fakta- ja dimensiotaulut saatiin muodostettua. Kuten aiemmin mainittiin, toimeksiantajan vaatimuksesta SQL-skriptejä ei esitetä. Tietovarastoa alettiin kehittämään toimeksiantajan palvelimelle Microsoft SQL Server 2012:een,

missä myös lähteinä käytetyt tietokannat sijaitsivat. Kun käyttäjätunnus SQL Serveriin oli luotu ja tarvittavat oikeudet annettu, voitiin ETL-prosessi aloittaa.

Aluksi luotiin uusi tyhjä tietokanta, joka toimisi tietovarastona. Jotta tietovarastoon päästiin hakemaan tietoa tauluihin, tarvittiin myös lukuoikeudet kirjanpidon ja toiminnanohjauksen tietokantoihin. ETL-prosessissa noudatettiin luvussa 2.2 esitettyä järjestystä, eli poiminta, muokkaus ja lataus. Ensimmäiseksi osa vanhan Transformer-kuution sisältämistä SQL-kyselyistä kopioitiin tietovarastoon, eli aloitettiin tiedon poimintavaihe.

Vanhasta kuutiosta pystyttiin hyödyntämään muutamia SQL-kyselyjä, joilla siihen oli tuotu tietoa. Näillä kyselyillä pystyttiin muodostamaan tietovarastoon taulut henkilödimensiolle, tilidimensioille ja, mikä tärkeintä, faktataulut kustannuksille sekä liikevaihdolle. Koska vanhan kuution kyselyt oli muodostettu suoraan IBM Cognos Transformerilla, ne olivat IBM:n Impromptu SQL -formaatissa, joka erosi hieman mm. funktioiltaan tavallisesta SQL:stä ja T-SQL:stä. Tämän vuoksi koodia jouduttiin muokkaamaan paljonkin SQL Serverille sopivampaan muotoon eli luvussa 2.1 mainittuun T-SQL:ään.

```
2
3   /* Impromptu SQL */
4   CASE WHEN (((cast_integer((od_left(Taulu2."nimi",6))))) IN (710,1378))
5   THEN (2)
6   ELSE ((cast_integer((od_left(Taulu2."nimi",6)))))
7   END
8
9   /*Sama käännettynä T-SQL:ään: */
10  CASE WHEN (((CAST((SUBSTRING(Taulu2.nimi,1,6))AS INT))) IN (710,1378))
11  THEN (2)
12  ELSE ((CAST((SUBSTRING(Taulu2.nimi,1,6))AS INT)))
13  END
```

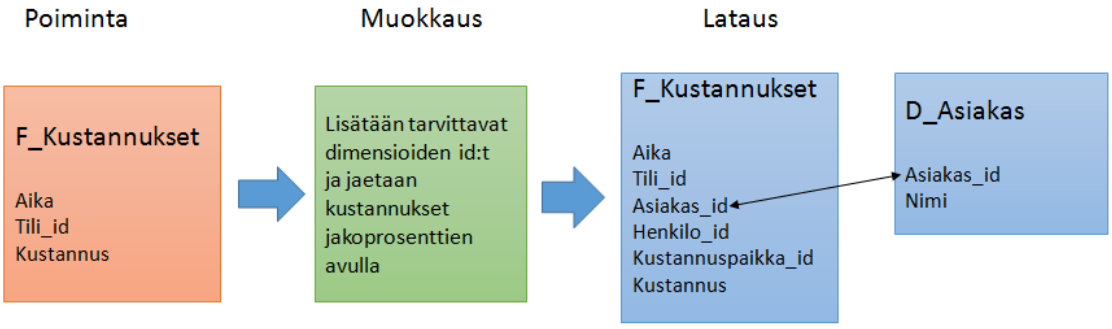
Kuva 6. Impromptun SQL-koodia ja sama T-SQL muodossa

Kuvassa 6 näkyy, kuinka funktiot voivat erota toisistaan SQL-kielen eri versioissa. Kuvassa käytetty CASE–lauseke on molemmissa sama (CASE-lause toimii hieman samalla periaatteella kuin IF-lause esim. C-kielessä), mutta sen sisällä käytetyt funktiot eroavat toisistaan. T-SQL:n CAST()-funktiolla muutetaan tiedon tyyppiä, joka on tässä tapauksessa INT (kokonaisluku). Impromptun koodissa

tämä tehdään suoraan cast_integer()-funktiolla. T-SQL:n SUBSTRING()-funktio palauttaa merkkijonosta halutun pätkän vasemmalta lukien. Sen parametreiksi asetetaan, monennenko merkin kohdalta lukeminen aloitetaan ja montako merkkiä luetaan. Impromptun od_left() lukee aina merkkijonon alusta alkaen niin monta merkkiä kuin parametriksi on määritelty. Myös pieniä kirjoitusmuodollisia eroja on havaittavissa kuvan esimerkissä.

Kun koodit oli saatu toimiviksi ja tiedot oli tuotu tietovarastoon tauluihin, päästiin tarkastelemaan syntyneiden taulujen sisältöä. Dimensiotauluista tarkastettiin, että ne sisältävät id-kenttien lisäksi tarvittavat lisätiedot. Faktatauluista tarkistettiin, minkä dimensioiden id-sarakkeita ne sisältävät eli mihin dimensioihin faktatiedot olivat jo allokoitu. Tässä vaiheessa kustannukset olivat allokoituna pelkästään ajan ja tilin mukaan (kuvan 7 poimintavaihe), joten allokointi asiakas-, henkilö- ja kustannuspaikkadimensioihin oli tehtävä erikseen luotettavan jakoperusteen avulla. Liikevaihto oli jo vanhassa kuutiossa saatu allokoitua ajan ja tilin lisäksi myös asiakkaittain, joten se täytyi vielä saada allokoitua henkilöihin ja kustannuspaikkoihin. Kuvassa 7 on poimintavaiheen kohdalla esimerkki kustannustaulun sisällöstä.

Entisessä kuutiossa oli myös jo osittain saatu allokoitua kustannuksia henkilöille muutamilta tileiltä (palkkatilit, matkakustannukset ym.), mutta ne jouduttiin noutamaan kaikki omina kyselyinään omiin tauluihinsa, koska osassa oli enemmän ja osassa vähemmän erilaisia sarakkeita. Tästä syystä nämä tiedot olivat hajallaan. Yksi vaihe oli yhdistää nämä tiedot samaan faktatauluun tietovarastossa. Näitä kustannuksia, joille henkilödimensio löytyi suoraan, kutsuttiin henkilökustannuksiksi. Nyt tietovarastoon oli siis tuotu faktataulut kustannuksista, henkilökustannuksista ja liikevaihdosta. Kun kaikki nämä vanhan kuution tiedot olivat nyt tietovarastossa, alettiin kehittämään ensimmäisenä henkilökustannuksille jakoperustetta, eli aiemmin mainittua prosentteja sisältävää summataulua, jolla allokaatio dimensioihin tehtäisiin. Kuvassa 7 on kuvattu ETL-prosessin kulkua ja nyt oli poimintavaihe faktataulujen osalta valmiina, joten ETL-prosessin muokkausvaihe aloitettiin.



Kuva 7. ETL-prosessin kulku.

Ensimmäiseksi mietittiin ja määriteltiin henkilökustannuksille luotettava jakoperuste, jolla ne saataisiin jaettua eri asiakkaiden ja kustannuspaikkojen kesken. Tässä päätettiin hyödyntää OlapConilla käytössä olevaa toiminnanohjauksen tuntikirjausjärjestelmää, josta voitiin laskemalla nähdä, kuinka paljon kukin henkilö oli käyttänyt aikaa eri asiakkaiden töihin. Tarvittavat tiedot tuntikirjauksista poimittiin toiminnanohjauksen tietokannasta. Tuntikirjausten kautta saatiin myös allokaatio kustannuspaikoille sen mukaan, minkä tyyppiselle työlle tuntikirjauksia on tehty. Kustannuspaikkoja määriteltiin neljä: konsultointi, koulutus, lisenssimyynti ja ylläpito. Tuntikirjausten perusteella voitiin laskea kuukausitasolle prosenttijakauma eri asiakkaiden kesken käytetyistä tunneista per henkilö ja kustannuspaikka. Muodostettiin kuukausitason summataulu, jossa nämä prosentit olivat.

Myyntipuolen työntekijöille jakoperuste täytyi toteuttaa eri tavalla, koska he eivät tee yleensä systemaattisesti tuntikirjauksia. Mietinnän jälkeen päätettiin toteuttaa summataulu, jossa on myyntihenkilöiden tekemät toimenpiteet asiakkaittain per kuukausi. Summataulua varten jokaiselle toimenpiteelle asetettiin kiinteät painoarvot numeroina, jotta saatiin keinotekoinen ”työaika” ja sen perusteella edelleen prosentuaalinen jakauma asiakkaiden kesken. Toimenpiteitä voivat olla esim. asiakastapaaminen tai tarjouksen tekeminen. Kustannuspaikkajakauma otettiin muiden työntekijöiden oletusjakauman perusteella, sillä myynnin toimenpiteillä ei välttämättä ole liitosta kustannuspaikkaan.

Kun kaikille työntekijöille oli saatu jakoprosentit summatauluihin, voitiin aloittaa henkilökustannuksien jakaminen asiakkaille ja kustannuspaikoille niiden avulla. Henkilökustannus-faktataulua muokkaamalla muodostettiin uusi Henkilökustannus-faktataulu, jossa henkilökustannukset olivat allokoituna aika-, tili- sekä henkilödimensioiden lisäksi kustannuspaikka-, ja asiakasdimensioille. Kaikki tarvittavat dimensiot olivat siis mukana. Vielä täytyi kuitenkin saada liiketoiminnan muut kulut Kustannus-taulusta allokoitua näille dimensioille ja sitä varten tehtiin henkilökustannusfaktataulun pohjalta vielä uusi prosenttitaulu, jossa oli kuukausittainen jakauma kaikille dimensioille. Tällä yleisjakaumalla saatiin haettua liiketoiminnan muut kulut ja allokoitua kaikille dimensioille. ETL-prosessin latausvaiheessa nämä tiedot ladattiin samaan faktatauluun ja näin saatiin yksi F_Kustannukset-taulu (kuvan 7 latausvaihe), jossa oli kaikki liiketoiminnan kustannukset ja tarvittavat dimensioavaimet. Kuvan 7 latausvaiheessa myös näkyy esimerkkinä, että asiakasdimensiotauluun on nyt linkitys, joten allokaatio on onnistunut.

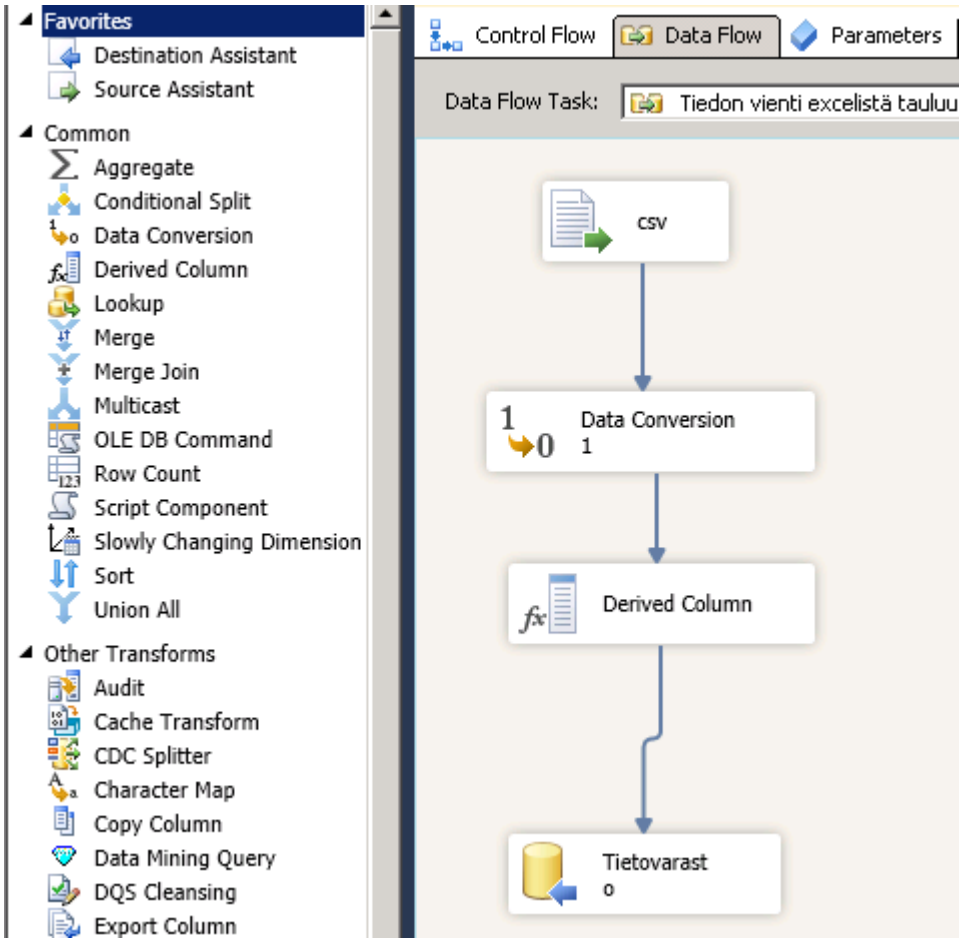
Liikevaihdon osalta ETL-prosessi oli huomattavasti nopeampi toteuttaa. Kuten aiemmin mainittiin, vanhan kuution SQL-kyselyssä liikevaihto allokoitui valmiiksi aika-, tili-, ja asiakasdimensioille. Faktatauluun täytyi siis vielä saada mukaan henkilö- ja kustannuspaikkadimensio. Kustannuspaikat saatiin kirjanpidon tietokannasta suoraan liikevaihtotositteelta. Lisenssimyynnin osalta liikevaihto kohdennettiin henkilödimensioon lisenssikaupan myyjälle ja loppu liikevaihto jaettiin kustannustenkin jakamisessa käytetyllä henkilökustannustaulusta saadulla jakaumalla henkilöille.

Tässä vaiheessa oli saatu valmiiksi molemmat faktataulut, mutta tietovarastoon tarvittiin vielä dimensiotaulut. Tilidimensiotaulu ja henkilödimensiotaulu oli tehty jo aiemmin vanhan kuution kyselyjä hyödyntäen, mutta vielä tarvittiin dimensiotaulut kustannuspaikoille, asiakkaille ja ajalle. Dimensiotaulut saatiin käytännössä suoraan toiminnaohjauksen tietokannasta ilman sen suurempia muokkauksia. Dimensiotaulujen osalta siis ETL-prosessin muokkausvaihe jäi käytännössä kokonaan pois, kun taulujen tiedot poimittiin sellaisenaan tietokannasta ja ladattiin suoraan tietovarastoon. Kun dimensiotaulut on kerran luotu, niitä ei yleensä tarvitse päivittää yhtä usein kuin faktatauluja, sillä dimensiotaulujen sisältö

muuttuu paljon harvemmin. Toisaalta asiakasdimensiotaulun sisältö muuttuu aina, kun uusia asiakkaita tulee, joten tämä täytyy ottaa huomioon ajojen ajastuksia tehdessä.

Tässä vaiheessa oli valmiina faktataulut kustannuksille ja liikevaihdolle, sekä dimensiotaulut ajalle, asiakkaille, henkilöille, kustannuspaikoille ja tileille. Taulujen sisältöä testailtiin vielä tekemällä erilaisia kyselyitä, jotta voitaisiin todentaa tiedon oikeellisuus. Testauksien jälkeen muutettiin kustannuksia niin, että ne jaettiin kahtia myyntikustannuksiin ja käyttökustannuksiin. Tämän avulla voitaisiin tarkastella myyntikatetta ja käyttökatetta erikseen, mikä oli myös työn tavoitteessa. Jako suoritettiin niin, että ensin määriteltiin myyntikustannuksiin kuuluvat tilit ja poimittiin niihin liittyvät kustannukset kustannustaulusta pois omaan faktatauluunsa. Jäljelle jääneet kustannukset olivat nyt käyttökustannuksia ja jako oli suoritettu.

Käyttökustannuksiin lisättiin vielä lopuksi lomapalkkojen jaksotukseen liittyviä kirjauksia erillissyötteenä Excel-tiedostosta. Tämä vaihe toteutettiin SSIS:llä. Ensin Excel muutettiin CSV-formaattiin, jotta se toimisi varmasti SSIS:n paketin tietolähteenä. Luotiin SSIS:ään uusi projekti, jossa tieto poimittiin CSV-tiedostosta, muokattiin tietovarastotaululle sopivampaan muotoon ja lisättiin tauluun. Projektista tallennettiin SSIS-paketti, joka sisälsi tämän prosessin ja se ajastettiin ajettavaksi tietovarastoon aiemmin tehtyjen SQL-proseduurien kanssa.



Kuva 8. SSIS-projekti

Kuvassa 8 on esitetty SSIS-projektin sisältöä. Vasemmalla näkyy osa ETL-prosessissa käytössä olevista työkaluista. Oikealla näkyy SSIS-paketin tiedon reitti tietovarastoon. Aluksi on poimittu tieto CSV-tiedostosta, minkä jälkeen sarakkeiden tietotyyppiä muutettiin merkkijonosta numeroksi (Data Conversion). Koska tietovaraston kohdetaulussa oli eri määrä sarakkeita kuin lähdetiedoissa, täytyi tietoihin lisätä vielä tarvittavat sarakkeet, jotta tieto saadaan sopimaan kohdetauluun. Samalla tehtiin myös yksi laskettu sarake (Derived Column). Lopussa valittiin tietovarastosta kohdetaulu, johon tieto siirretään. Paketin toimivuuden testauksen jälkeen projekti tallennettiin ja paketti otettiin talteen myöhempää ajastusta varten.

ETL-prosessi alkoi nyt olla valmiina ja vielä testailtiin tietovaraston tauluja, jotta niissä ei olisi virheitä. Uusien faktataulujen lukuja summailtiin ja verrattiin entisiin lukuihin ja voitiin todeta, että luvut täsmäävät eivätkä ole muuttuneet vir-

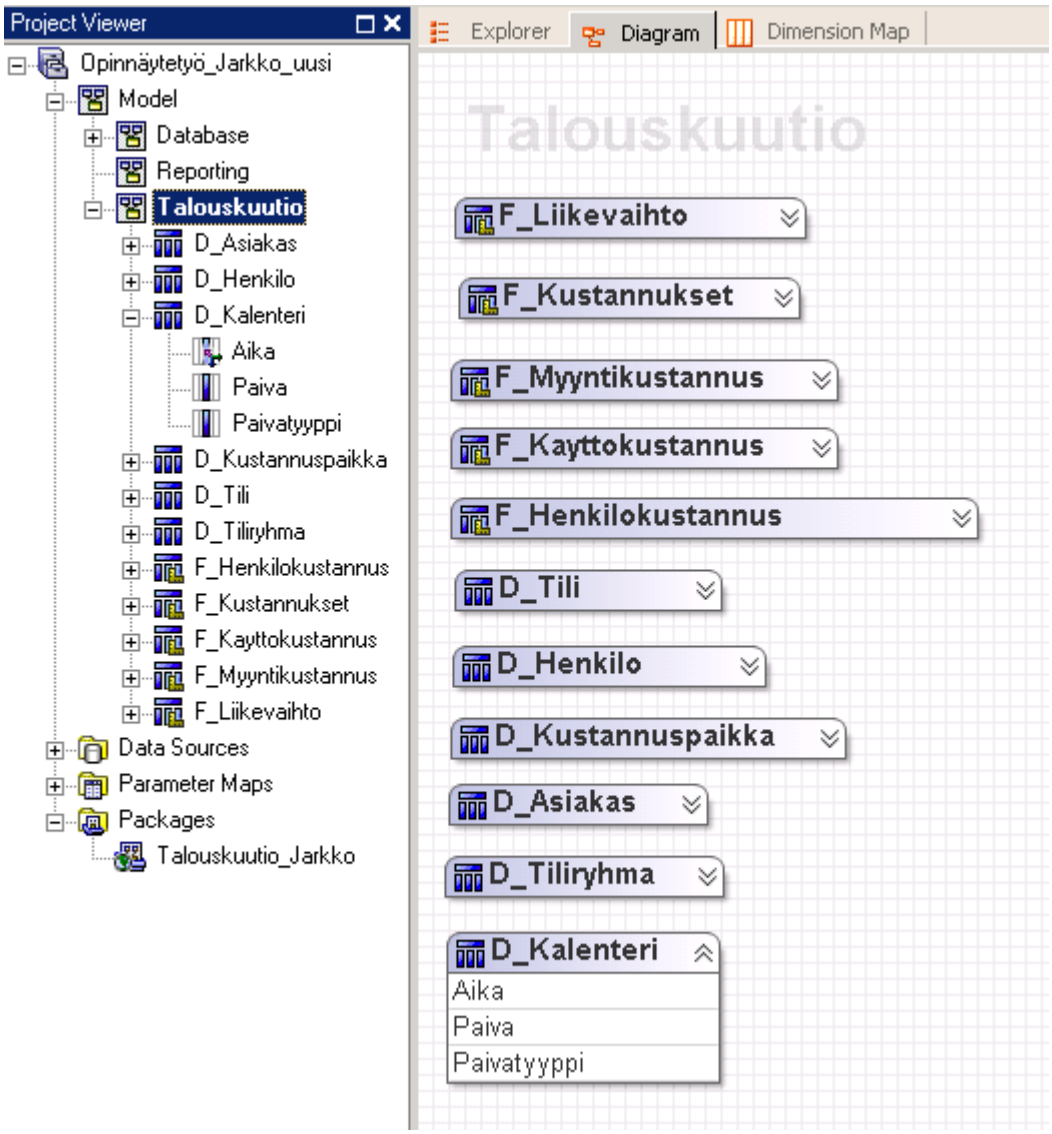
heellisiksi ETL-prosessin muokkausvaiheessa. Kun tietovarasto alkoi olla valmiina, työssä voitiin siirtyä Cognos-palvelimelle työstämään BI-ratkaisua IBM Cognoksen tuotteilla.

5.3 Framework-paketin toteutus

IBM Cognos Framework Managerilla toteutettiin ja julkaistiin Cognos-portaaliin paketti, jota käytettiin IBM Cognos Transformerissa kuution tietolähteenä. Aluksi Framework Managerilla luotiin uusi projekti. Projektissa luotiin yhteys tietokantapalvelimella sijaitsevaan uuteen tietovarastoon ja lisättiin se projektin tietolähteeksi. Seuraavaksi pystyttiin hakemaan tietovaraston sisältämät taulut Framework-projektiin, jonka jälkeen taulujen sisältämien sarakkeiden käyttötyyppejä alettiin muokkaamaan. Framework Managerissa sarakkeen tyyppi voi olla id, fakta (Fact) tai määrite (Attribute). Kaikki dimensioiden id-kentät asetettiin id:ksi ja faktataulujen sisältämät lukuarvot, eli esim. kustannusten euromäärät, faktoiksi. Dimensiotauluissa id:n takana olevat lisätiedot asetettiin määritteiksi. Käyttötyyppien asettaminen kannattaa tehdä huolella, koska sillä on vaikutusta tietojen esiintymiseen raporteilla. Esimerkiksi, kun luvun tyyppinä on fakta, Cognoksen raportointivälineet osaavat käsitellä sitä oikein mm. summaamalla arvoja. Jos luvun tyyppinä on määrite, sitä käsitellään kiinteänä lisätietona, jota ei summata raporteilla.

Projektiin luotiin kolme nimitilaa: Database, Reporting ja Talouskuutio. Database nimitilan alla oli kaikki tietovaraston taulut ilman muokkauksia. Reporting nimitilaan siirrettiin myös kaikki tietovaraston taulut, joihin oli asetettu käyttötyypit ja taulujen välille luotiin myös suhteet perusraportointia varten. Framework Managerilla suhteet fakta- ja dimensiotaulujen välille saatiin luotua kätevästi graafisella käyttöliittymällä, jossa suhteiden luonti tapahtuu linkittämällä taulujen id-kenttiä toisiinsa, eli samalla tavalla kuin aiemmin luvussa 2.2. kerrottiin. Talouskuutio-nimitilaan tuotiin Database-nimitilasta vain ne taulut, joita tarvittiin tuoda kuutioon, eli dimensio-, ja faktataulut. Erilaiset aputauluina käytetyt taulut, kuten prosenttitaulut, jätettiin pois. Kuutiota varten ei tarvinnut luoda taulujen välisiä suhteita, sillä se toteutetaan eri tavalla IBM Cognos Transformerilla.

Muutamia taulujen ja sarakkeiden nimiä käytiin muuttamassa tässä vaiheessa vielä tietovaraston puolella paremmiksi, koska huomattiin, että nimet voisivat olla kuvaavampia. Kun muutokset oli tehty ja kaikki tarvittava tieto oli saatu kerättyä, todennettiin vielä tietojen oikeellisuus testaamalla tauluja ja sitten luotiin paketti, joka julkaistiin IBM Cognos Connection -portaaliin. Koska tässä työssä oli tarkoitus tehdä kuution pohjalta analyysia, otettiin pakettiin mukaan vain Talouskuutio-nimitilan sisältö. Jos joskus tulee tarvetta tehdä perusraportointia tämän tietovaraston pohjalta, voidaan käyttää Reporting-nimitilan tietoja, joihin on asetettu taulujen väliset suhteet ja tietojen käyttötyypit ja julkaista niistä oma pakettinsa. Kun paketti on kerran julkaistu, sitä ei tarvitse julkaista enää uudelleen, ellei siihen tarvitse lisätä uusia tietoja mukaan. Paketti on aina ajan tasalla, sillä sen kautta ollaan yhteydessä tietovarastoon.



Kuva 9. Näkymä Framework Manager -projektista

Kuvan 8 Framework Manager-näkymässä on vasemmalla projektin sisältö, eli malli (Model) nimitiloineen, tietolähteet (Data Sources), parametrit (Parameter Maps), joita ei tässä toteutuksessa tarvittu, ja paketit (Packages). Oikealla on Talouskuutio-nimitilan sisältö graafisessa käyttöliittymässä, jossa taulujen sisältämiä sarakkeita voidaan tarkastella ja muokata, sekä tehdä taulujen välisiä suhteita id-sarakkeita linkittämällä. Kuten aiemmin mainittiin, kuutiota varten ei tehty taulujen välisiä suhteita, joten kuvassakaan niitä ei näy. Kuvassa on avattuna aikadimensiotaulun sisältö (D_Kalenteri). Tämän dimensiotaulun id-sarakkeena on Aika ja kaksi muuta saraketta ovat määritteitä. Erityyppiset sarakkeet on kuvattu erilaisilla kuvakkeilla, kuten myös fakta- ja dimensiotaulut.

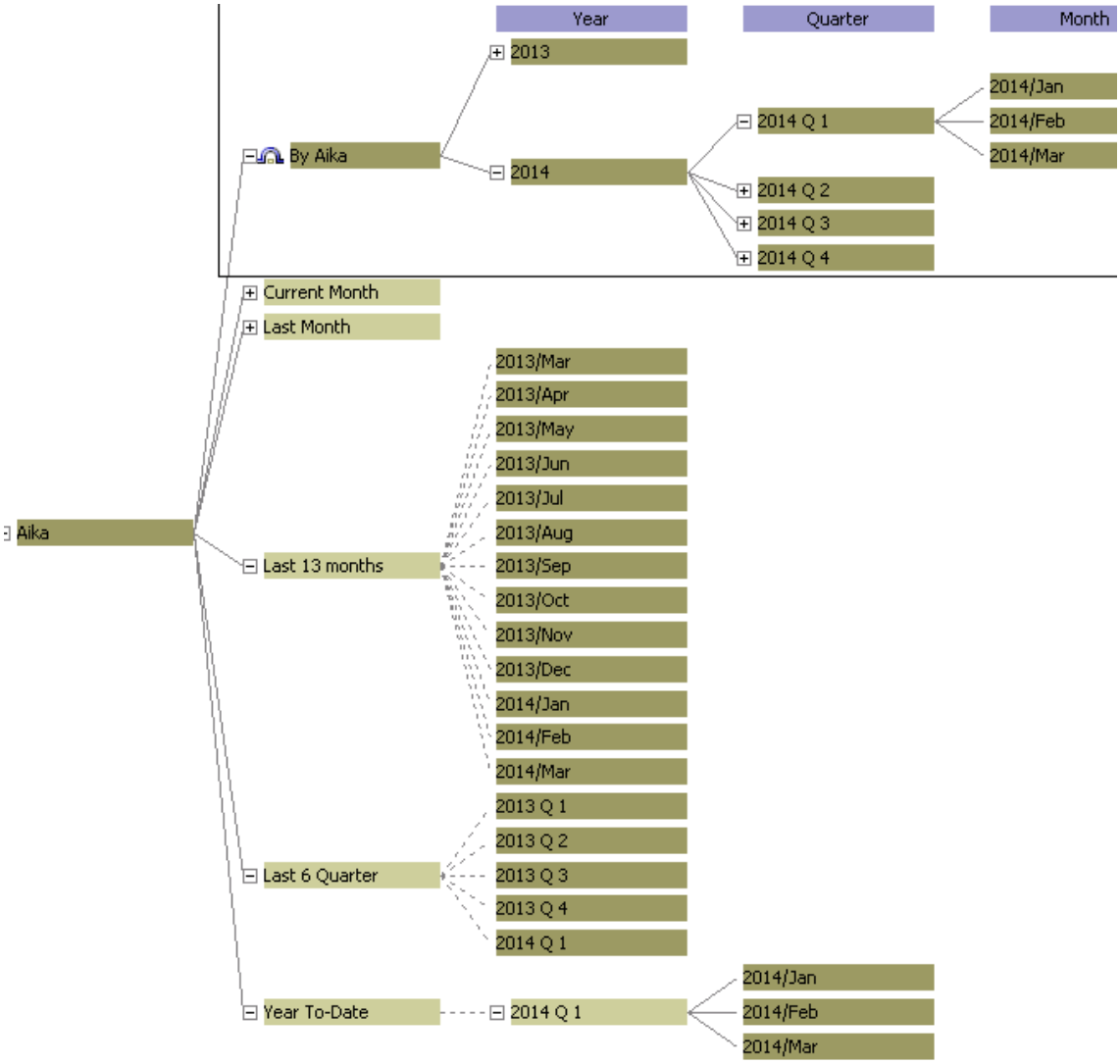
Framework Manager tunnistaa faktatauluksi sellaiset taulut, joissa jokin sarake on asetettu tyypiltään faktaksi.

Framework Managerilla projektia tehdessä huomattiin, kuinka kätevästi paketti syntyy, kun tietovarastotasolla on taulujen ja sarakkeiden nimet valmiiksi asetettu selkeiksi. Jos nimeämiset olisivat olleet epäselviä tai epäjohdonmukaisia, olisi Framework Managerissa jouduttu vielä nimeämään tauluja ja sarakkeita uudelleen, jotta ne soveltuisivat paremmin BI-käyttöön. Kun talouskuutiopaketti oli saatu julkaistua Cognoksen portaaliin Framework Managerista, käytiin vielä tarkastamasta portaalista, että paketti oli oikeassa kansiossa ja että siihen oli käyttöoikeudet ainoastaan johtoryhmälle. Seuraavaksi paketin pohjalta aloitettiin toteuttamaan kuutiota IBM Cognos Transformerilla.

5.4 OLAP-kuution toteutus Transformerilla

Asensin ensin omalle työkoneelleni IBM Cognos Transformerista 7-version, jolla tutkin vanhaa kuutiota, sillä siitä sai hyvää mallia uuden kuution toteutukseen. Siitä näki mm. kuinka mittarilukujen muotoilut oli asetettu ja miten dimensioita oli rakenneltu. Kun olin kunnolla perehtynyt vanhan kuution toteutukseen, aloitin uuden kuution toteuttamisen Transformerin 10-versiolla.

Ensimmäisenä luotiin Transformeriin uusi projekti, johon asetettiin aiemmin tehty Talouskuutiopaketti tietolähteeksi Cognos-portaalista. Seuraavaksi tuotiin tietolähteestä tauluja projektiin. Aika-, asiakas-, henkilö-, kustannuspaikka- ja tilidimensiotaulut tuotiin ensimmäiseksi ja sitten vielä faktataulut: myyntikustannus, käyttökustannus ja liikevaihto. Transformer osaa luoda automaattisesti esim. aikadimension kuutioon dimensiotaulun pohjalta. Tätä ominaisuutta hyödynnettiin ja saatiin jo yksi dimensio kuutioon lähes valmiiksi. Aikadimension hierarkia muodostui ylätasosta (kaikki ajat), vuodesta, kvartaalista ja kuukaudesta. Kuten luvussa 2.3 mainittiin, hierarkia mahdollistaa porautumisen, eli aikadimension ylimmältä tasolta, jossa on kaikki ajat mukana voidaan porautua vuositasolle, siitä edelleen valitun vuoden kvartaaleille ja vielä edelleen sen kvartaalin kuukausille. Kuvassa 10 on esitetty kuution aikadimension sisältö.

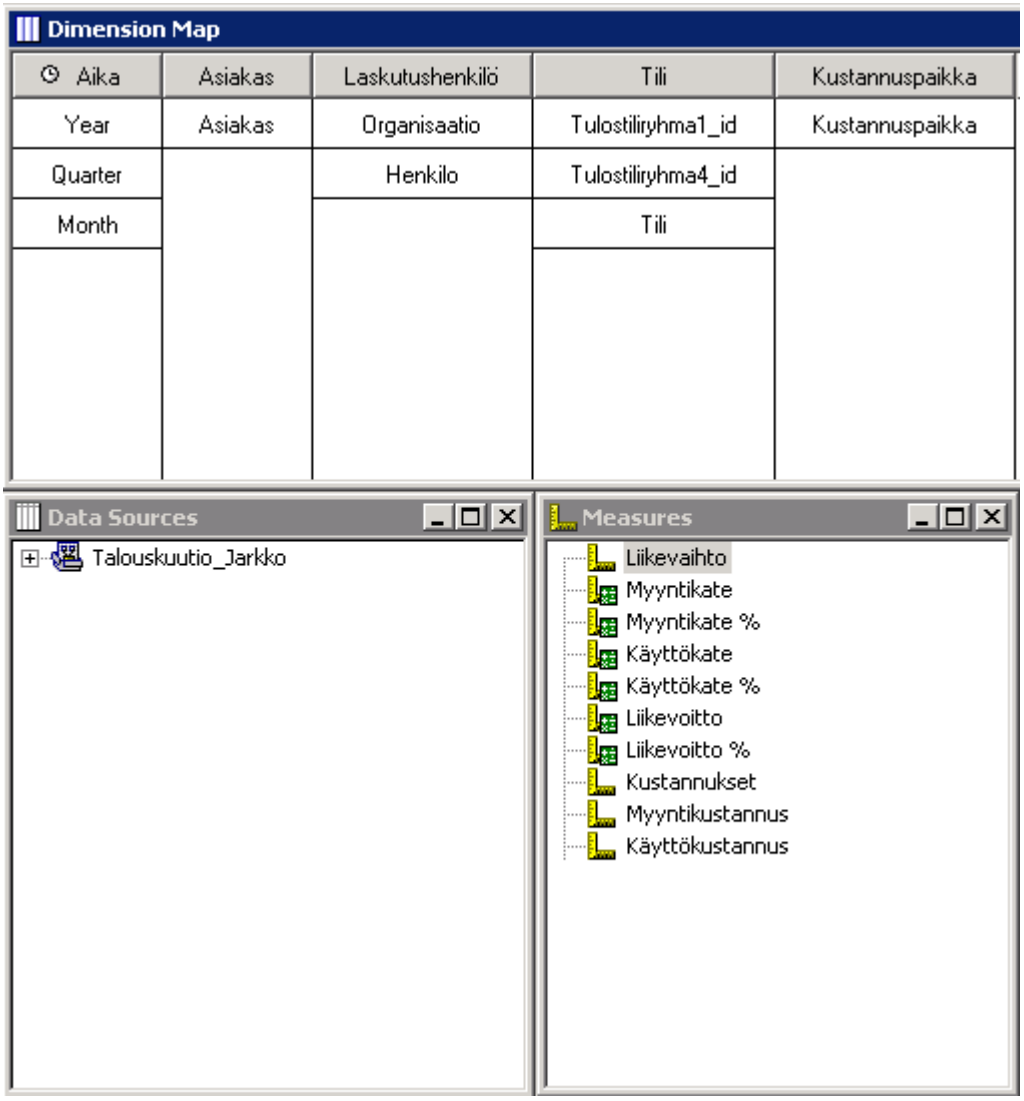


Kuva 10. Aikadimension hierarkiatasot.

Toimeksiantaja määritteli myös aikadimensioon tarpeellisia lisätasoja, joita olivat Last 13 months, Last 6 Quarters ja Year To-Date. Nämä ovat näkyvillä kuvassa 10. Last 13 months sisältää viimeiset 13 kuukautta nykyiseen kuukauteen asti, Last 6 Quarters taas sisältää vastaavasti 6 viimeisintä kvartaalia ja Year To-Date kuukaudet kuluvan vuoden alusta nykyiseen kuukauteen asti. Nämä lisätasot mahdollistavat tietojen monipuolisemman tarkastelun aikaa vasten. Kuvassa 10 näkyvät Current Month ja Last Month ovat Transformerin automaattisesti määrittelemiä.

Muut dimensiot määriteltiin manuaalisesti. Asiakasdimensioon tehtiin vain yksi taso, Asiakas, sillä asiakkaita ei ollut tarvetta kategorisoida mihinkään ryhmiin, joista olisi voitu tehdä hierarkiaa. Henkilödimension nimeksi tuli kuvaavammin Laskutushenkilö ja siinä oli kaksi tasoa: organisaatio ja henkilö. Organisaatiotasolla oli johtoryhmä, hallitus ja muu henkilöstö erikseen ja tämän tason alta henkilötasolta löytyi itse henkilöt. Tilidimensioon tehtiin kolme tasoa: Tiliryhmä1, jonka alla oli vielä toinen tarkempi tiliryhmä ja sen alla itse tilit. Kustannuspaikkadimensioon tehtiin vain yksi taso, jossa oli kustannuspaikat, koska kustannuspaikkojakaan ei ollut tarve ryhmitellä mitenkään. Dimensioiden tekeminen oli todella kätevää Transformerin graafisella käyttöliittymällä, sillä siinä pystyttiin hiirellä drag-and-drop–periaatteella luomaan tietolähteestä tuotujen taulujen sarakkeista tasoja dimensioihin. Samalla tavalla määriteltiin kuution mittariluvut, eli ne vain vedettiin tuoduista tauluista Measures–kenttään.

Kuution mittareiksi asetettiin faktatauluista liikevaihto, käyttökustannus ja myyntikustannus. Näiden pohjalta tehtiin lisäksi laskettuja mittareita, joita olivat myyntikate, myyntikate %, käyttökate, käyttökate %, liikevoitto ja liikevoitto %. Myyntikate saatiin vähentämällä liikevaihdosta myyntikustannukset ja käyttökate samalla tavalla vähentämällä liikevaihdosta käyttökustannukset. Liikevoitto taas saatiin, kun vähennettiin kaikki kustannukset liikevaihdosta. Prosenttimittareiden laskemiseen käytettiin Transformerin percent()-funktiota, joka laskee yhden luvun prosentin toisesta luvusta, esimerkiksi percent(”käyttökate”, ”liikevaihto”). Kun kuutioon oli luotu tarvittavat mittarit, määriteltiin vielä niiden formaatit, mm. kuinka monta desimaalia luvuissa on ja ovatko ne prosentteja vai euroja.



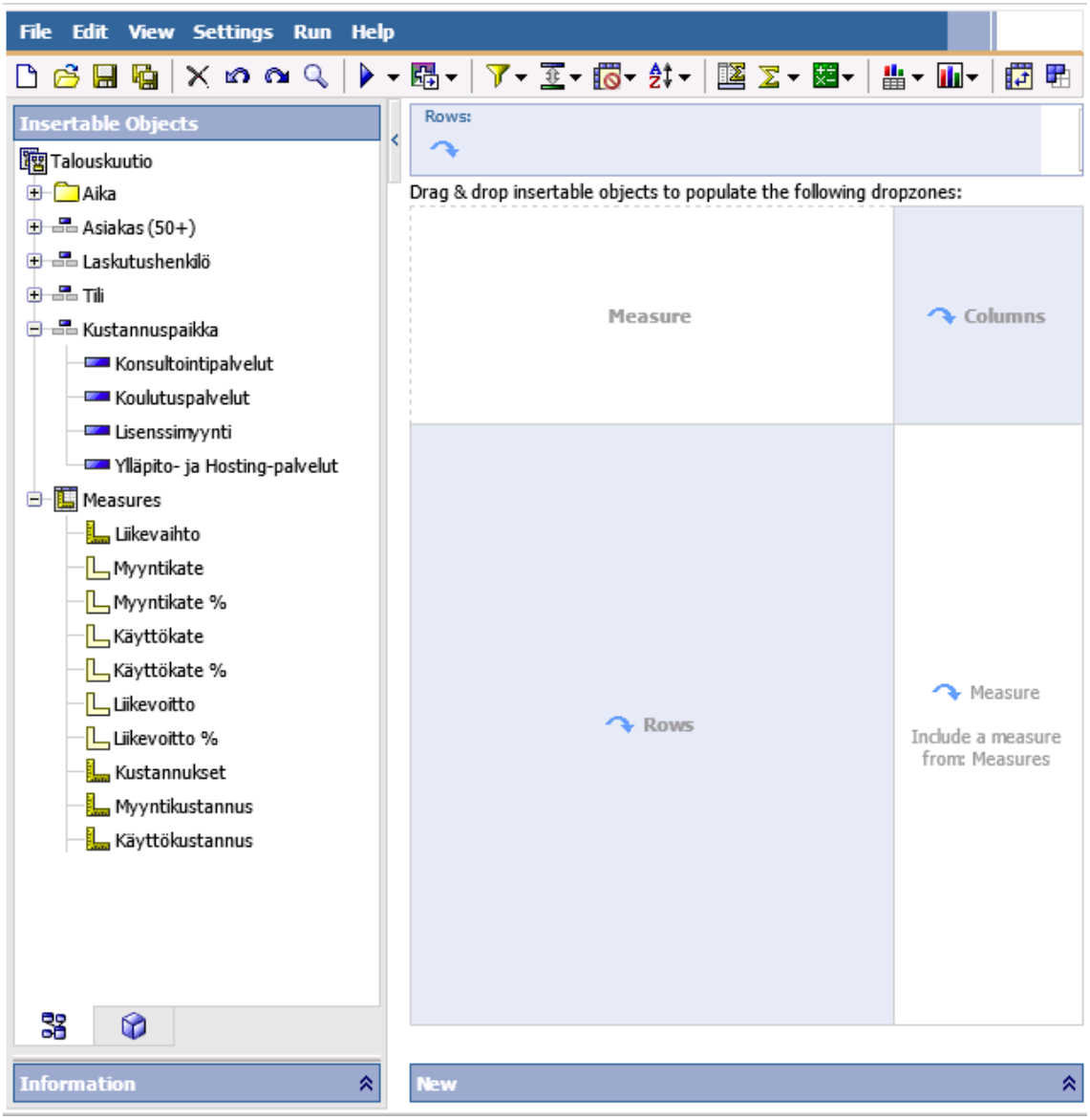
Kuva 10. OLAP-kuution malli Transformerissa.

Kuten kuvassa 10 näkyy, kuutiossa olivat nyt kaikki dimensiot ja niiden sisältämät hierarkiat sekä tarvittavat mittariluvut. Vielä täytyi tarkistaa, että kaikki mittariluvut yhdistyvät kaikkien dimensioiden jokaiselle tasolle. Mittarit ja dimensiot allokoituvat toisiinsa automaattisesti, jos kaikissa tauluissa on id-sarakkeet nimetty täsmälleen samalla tavalla, eli esimerkiksi asiakastaulussa on id-kentän nimi oltava sama kuin faktatauluissa. Tämän jälkeen Transformer osaa yhdistää tiedot automaattisesti. Tässä vaiheessa huomattiin, että muutamien kenttien nimet eivät täsmänneetkään, joten käytiin muuttamassa niitä SQL Serverillä tietovarastossa, jonka jälkeen nekin saatiin allokoitumaan kaikkiin dimensioihin. Nimien muutoksen olisi voinut tehdä Transformerissakin, mutta lähtökohtaisesti on aina parempi, että tietovarastotasolla on kaikki kunnossa.

Seuraavaksi täytyi mallin pohjalta luoda itse kuutio. Kuutiota luodessa otettiin versiointi käyttöön, eli viisi viimeisintä versiota kuutiosta säilytetään palvelimella niille tarkoitetussa kansiossa. Tämän avulla voidaan kuutiosta ottaa vanhempi toimiva versio käyttöön, jos joskus kuution päivityksen jälkeen huomataankin, ettei se toimi enää oikein. Kuutiolle annettiin sopiva nimi, jonka jälkeen Transformer loi sen palvelimelle tiedostoksi ja se julkaistiin Cognos-portaaliin raportointityökalujen lähteeksi. Tarkistettiin portaalista vielä, että kuutionkin käyttöoikeudet ovat pelkästään johtoryhmällä. Nyt kuution pohjalta voitiin tehdä moniulotteista analyysiä IBM Cognos Analysis Studio -tuotteella.

5.5 Kuution tarkastelu Analysis Studiolla

Analysis Studio avattiin IBM Cognos Connection -portaalista, josta se avautui omaan selainikkunaansa. Tietolähteeksi valittiin juuri julkaistu kuutio. Analysis Studiossa näkyi kuution sisältö, eli dimensiot (aika, asiakas, laskutushenkilö, tili, kustannuspaikka) ja mittariluvut. Oletuksena Analysis Studion raporttinäkymässä on tyhjä ristiintaulukko, johon voidaan hiirellä raahaamalla tuoda dimensioita ja mittareita riveille ja sarakkeisiin. Kuvassa 9 on Analysis Studion oletusnäkymä, jossa kustannuspaikkadimension sisältö ja mittariluvut ovat näkyvillä.



Kuva 9. Analysis Studio-näkymä

Ristiintaulukon/matriisin lisäksi tai sen tilalle voidaan tehdä myös kuvaavia diagrammeja ja graafeja, joista Analysis Studio tarjoaa monia erilaisia vaihtoehtoja. Tässä työssä kuitenkin keskityttiin tarkastelemaan lukuja taulukossa ja aivan ensimmäisenä tutkittiin, olivatko mittariluvut oikein. Suurimmaksi osaksi kustannukset ja liikevaihto täsmäsivät, kun verrattiin oikeaksi tiedettyihin arvoihin, mutta joidenkin tilien osalta löytyi myös heittoja. Tässä vaiheessa siirryttiin takaisin ETL-prosessissa toteutettuihin SQL-proseduureihin etsimään, mistä tämä johtui ja tekemään tarvittavia muutoksia, jotta luvut saataisiin näkymään oikein. Yleensä huomattiin että SQL-koodiin asetetut suodatukset jättivätkin myös tar-

peellisia lukuja tuomatta, joten niitä jouduttiin vielä tarkentamaan. Kuutio luotiin aina uudelleen, jos tietovarastoon tehtiin muutoksia, jotta se olisi ajan tasalla.

Jokaista mittaria analysoitiin jokaista dimensiota vasten ja kun voitiin todeta, että kuutio toimii hyvin, työ alkoi olla lopuillaan. Koska käytössä oli kuutio, tietojen käsittely ja porautuminen oli todella nopeaa, sillä kuten luvussa 2.3 mainittiin, kuutiossa mittariluvut on summattu valmiiksi jokaisen dimension jokaista tasoa vasten. Tietojen analysointi olisi ollut huomattavasti hitaampaa, jos kyselyitä olisi tehty suoraan tietovarastoa vasten, koska silloin esim. tietojen summaaminen tapahtuu aina kyselyhetkellä. Analysis Studiolla ei ollut tarvetta tehdä varsinaisia valmiita raporttinäkymiä, sillä toimeksiantaja voi itse tehdä nopeasti sellaisia näkymiä kuin milloinkin tarvitsee.

5.6 Työn lopputoimet

Tietovarastointi ja BI-prosessi oli nyt viety päätökseen ja toimeksiantaja tutki vielä lopputulosta Analysis Studiolla. Tässä vaiheessa toimeksiantajalta tuli vielä muutosvaatimuksia, joista osa oli kuution käytettävyyteen liittyviä viilauksia, esimerkiksi dimensioiden ja mittareiden järjestys, kuinka ne näkyvät Analysis Studiossa ja Transformerin luomien turhien aikadimension tasojen piilotus pois näkyvistä. Muita muutoksia olivat myös tietovarastotasolla tehtävät viilaukset mm. muutamien tilien kustannusten allokointiin liittyen. Esimerkiksi sellaisten tilien kustannukset, joiden voitiin suoraan sanoa kohdentuvan pelkästään johtoryhmään, täytyi allokoida uudelleen henkilödimensiolle niin, että kustannukset jakautuivat vain johdon henkilöille. SQL Serverillä käytiin tekemässä tarvittavat muutokset proseduureihin, jonka jälkeen Transformerilla hienosäädettiin kuution sisältöä vaatimuksien mukaan käyttäjäystävällisemmäksi ja sitten luotiin kuutio uudelleen. Analysis Studiolla testailun jälkeen voitiin todeta, että nyt kuutio täyttää vaatimukset.

Viimeinen vaihe työn toteutuksessa oli proseduurien ja SSIS-paketin ajastaminen SQL Serveriin, sekä OLAP-kuution päivityksen ajastaminen. SQL Serveriin ajastukset tehtiin SSMS:llä. Proseduurit asetettiin ajettavaksi oikeassa järjes-

tyksessä yöaikaan sen jälkeen, kun lähdejärjestelmien tietokannat ovat päivittyneet. Ajastuksessa tehtiin SQL Serverille uusi työ (Job), joka sisälsi proseduurien ajot vaiheina (Steps) ja työlle asetettiin kellonaika, jolloin SQL Server tulee käynnistämään sen. Kuution ajastaminen tapahtui Cognos-palvelimen Task Schedulerilla, joka on osa Windows Serveriä. Scheduleriin ajastettiin skripti, joka loi halutun kuution. Skripti oli valmiina, koska palvelimella oli ajastettu jo muitakin kuutioita vastaavalla tavalla. Ajastusten jälkeen tietovaraston ja OLAP-kuution käyttäjillä oli aina automaattisesti ajankohtaiset tiedot käytössään.

6 Tulokset

Opinnäytetyön tuloksena syntyi BI-ratkaisu, jolla toimeksiantaja voi tutkia yrityksen liiketoiminnan tulosta useita eri dimensioita vasten. Ratkaisuun sisältyi uusi tietovarasto toimeksiantajan palvelimelle Microsoft SQL Server 2012:een. Tietovaraston pohjalta toteutettiin Framework Manager-paketti toiselle palvelimelle IBM Cognos Connection-portaaliin. Framework Manager–paketin pohjalta toteutettiin OLAP-kuutio, jossa dimensioina olivat aika, asiakas, laskutushenkilö, tili ja kustannuspaikka. Kuution mittarilukuina olivat liikevaihto, myyntikate, myyntikate-%, käyttökate, käyttökate-%, liikevoitto, liikevoitto-%, kustannukset, myyntikustannus ja käyttökustannus. Kuutio on julkaistu tietolähteeksi IBM Cognos Connection-portaaliin ja sitä voidaan käyttää mm. Analysis Studiolla nopean moniulotteisen analysoinnin tekemiseen.

Tulokset vastaavat sitä, mitä tavoitteeksi oli asetettu. Vanha OLAP-kuutio saatiin nyt korvattua uudella kuutiolla, jossa asiakaskohtaisen katteen tarkastelu on mahdollista. Toteutuksessa otettiin huomioon myös käyttökate ja myyntikate erikseen ja kuutiossa on mukana kaikki vaaditut mittarit ja dimensiot. Tuloksena syntyneen kuution lisäksi opinnäytetyön ETL-prosessivaiheessa saatiin selville, kuinka asiakaskohtaisen katteen laskenta saadaan tarpeeksi luotettavalle tasolle, jotta sitä voidaan käyttää hyödyksi päätöksenteossa. Luotettavat jakoperus-

teet kustannuksille saatiin toimintolaskentaa hyödyntäen erilaisista suoritteista, kuten tuntikirjauksista ja myyjien tekemistä toimenpiteistä eri asiakkaille.

7 Pohdinta

Tietovarastointi ja BI-prosessit toteutettiin hyvin pitkälti samoilla tavoilla kuin opinnäytetyön tietoperustassa on esitetty. Lähestymistapana oli yleisimmin käytetyt tekniikat, kuten tietovaraston rakentaminen tähtimallin mukaiseksi ja ETL-prosessissakin noudatettiin kaavaa, jossa tieto poimitaan, muokataan ja lopuksi ladataan tietovarastoon. ETL-prosessin toteutus ohjelmoiduilla SQL-proseduureilla ETL-välineen sijasta tuntui kätevimmältä tämän työn kannalta, koska vanhan kuution SQL-kyselyitä pystyttiin hyödyntämään osittain uudessa toteutuksessa. Halusin myös oppia SQL-kielen käyttöä ja nähdä syvemmin, kuinka itse koodit toimivat. ETL-työkalulla tehtiin kuitenkin Excel-tietolähteen sisällön tuominen tietovarastoon, joten molempia ETL-prosessin toteutusvaihtoehtoja hyödynnettiin. Erot kahden toteutustavan välillä olivat suuret, koska ETL-työkalulla tehdessä ei välttämättä tarvitse olla SQL-koodin kanssa missään tekemisissä, vaan kaikki tehdään graafisesti. OLAP-kuutiota tehdessä otettiin huomioon luvussa 2.3 mainitut suositukset dimensioiden määrästä. Kuutio pysyi siistinä ja helppokäyttöisenä, kun siihen tuotiin sopiva määrä dimensioita ja mittarilukuja.

Eniten aikaa työn toteutuksessa kului ETL-prosessiin, sillä lähdetaulujen sisältöjen selvitykseen kului aikaa, kun ensimmäisenä täytyi saada hyvä kuva siitä, mistä mitäkin tietoa haetaan. ETL-prosessin muokkausvaiheessa apuna käytettyjen prosenttitaulujen toteutuksessa tuli myös paljon kokeiluja, jotka eivät toimineetkaan niin kuin ajattelin ja niiden hiomiseen kului aikaa. Tietovaraston lukuja täytyi myös verrata tarkasti lähdekantojen tietoihin, jotta raportoinnissa tarkasteltavat tiedot olivat varmasti oikeita. Yhtenä päänvaivana oli myös, että lähdetietokantojen välillä oli eroja collation-asetuksissa, eli siinä, mitä merkistöjä ne tukevat. Tämä asia vaati selvittelyä, kun kaikki tiedot eivät sopineetkaan yhteen.

Esimerkiksi skandinaaviset merkit eivät toimi jos collation-asetus ei ole oikea ja merkkikokoriippuvuus (case-sensitivity) voi olla päällä joissain collation-tiloissa. Kun tiedot yhdistettiin tietovarastossa, niin collation täytyi muistaa asettaa kaikille tiedoille samaksi. Kaiken kaikkiaan työ kuitenkin sujui ilman suurempia ongelmia ja työn tekeminen eteni sujuvasti vaiheesta toiseen. Työlle asetetussa aikataulussa pysyttiin hyvin.

Opinnäytetyön toteutuksessa opin paljon uusia asioita, varsinkin tietovarastoinnista ja Business Intelligencestä. SQL-kieli tuli paljon tutummaksi työn ansiosta ja nyt osaan hyödyntää sitä paljon tehokkaammin ja monipuolisemmin ja ymmärrän sen syntaksia paremmin. BI:n puolesta työ taas opetti, minkälaista tietoa yrityksissä saatetaan haluta tarkastella ja millä tavoin. Tutuksi tuli koko tietovarastointiprosessi alusta loppuun: suunnittelusta ETL-vaiheeseen ja siitä eteenpäin BI-ratkaisujen toteutukseen. Aiemmasta harjoittelustani OlapConilla oli hyötyä työn toteutuksessa, sillä kaikki käytetyt ohjelmistot Transformeria lukuunottamatta olivat ainakin osittain tuttuja jo ennestään.

Opinnäytetyössä saavutettiin kaikki lähtötilanteessa asetetut tavoitteet ja tuloksena syntynyt BI-ratkaisu tuli toimeksiantajayrityksen johdon käyttöön ja se toimii päivittäisen päätöksenteon tukena. Käytön alkuvaiheessa liikevaihdon ja kustannusten lukuja vertaillaan vielä oikeisiin kirjanpidon lukuihin, sillä kuutio on mm. asiakasmäärältään niin laaja, että sen lukujen täydellinen testaaminen tulee viemään aikaa. Ylläpitotöitä voi ilmaantua, jos on tarve esim. muuttaa lukujen allokaatiota dimensioille joidenkin tilien osalta tai jos ajastetuissa ajoissa sattuu jokin virhe. Virhetilanteessa toimeksiantaja voi ottaa käyttöön kuutiosta tehdyt varmuuskopiot, joten toimiva kuutio on aina saatavilla. Jatkokehitysmahdollisuuksia työlle löytyy useita, sillä vaikkapa tietovarastoon voidaan tuoda lisää tietoa jostain muusta aihealueesta ja katelaskentaa saatetaan haluta ulottaa muihinkin liiketoiminnan osa-alueisiin. Opinnäytetyössä tehtyä Framework-pakettia voidaan käyttää jatkossa perusraportoinnin, dashboardien, ym. BI-ratkaisujen tekemiseen, kun taulujen väliset suhteetkin ovat jo valmiina.

Lähteet

1. Hovi, A., Hervonen, H. & Koistinen, H. Tietovarastot ja Business Intelligence. Porvoo. 2009. 196 s. ISBN: 978–951-0-34792-8.
2. Hovi, A., Ylinen, J. & Koistinen, H. Tietovarastot liiketoiminnan tukena. Jyväskylä. 2001. 276 s. ISBN: 951–762-777-7.
3. Krmac, Evelin V. Intelligent Value Chain Networks: Business Intelligence and Other ICT Tools and Technologies in Supply/Demand Chains. Teoksessa: Renko, S. Supply Chain Management - New Perspectives. [Verkkojulkaisu]. 2011. 770 s. [Viitattu 2.4.2014]. Saatavissa: http://cdn.intechopen.com/pdfs-wm/18527.pdf. ISBN: 978–953-307-633-1.
4. ISO/IEC 9075-1:2011. Information technology -- Database languages -- SQL -- Part 1: Framework (SQL/Framework). 2011. [Viitattu 14.5.2014]. Saatavissa: http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=53681
5. Joan, Ben. Difference between SQL and T-SQL. 2011. [Viitattu 10.4.2014]. Saatavissa: http://www.differencebetween.net/technology/software-technology/difference-between-sql-and-t-sql/.
6. Goldstein, Jackie. Writing SQL Queries: Let's Start with the Basics. 2005. [Viitattu 10.4.2014]. Saatavissa: http://technet.microsoft.com/en-us/library/bb264565(v=sql.90).aspx.
7. Houston, Neal. Business Intelligence 101 – A Beginner’s Guide to BI Software. 2011. [Viitattu 3.4.2014]. Saatavissa: http://plotting-success.softwareadvice.com/beginners-guide-to-bi-software-1113011/.
8. Rouse, Margaret. Star Schema. 2010. [Viitattu 4.4.2014]. Saatavissa: http://searchdatamanagement.techtarget.com/definition/star-schema.
9. Lane, Paul. Oracle Database Data Warehousing Guide, 10g Release 2. [Verkkojulkaisu]. 2005. Saatavissa: http://docs.oracle.com/cd/B19306_01/server.102/b14223.pdf.
10. IBM. 2014. Ranking sample. [Viitattu 12.4.2014]. Saatavissa: http://pic.dhe.ibm.com/infocenter/cbi/v10r1m1/index.jsp?topic=%2Fcom.ibm.swg.ba.cognos.ug_mig.10.1.1.doc%2Fc_smp_ranking.html.
11. Microsoft Corporation. 2014. Features Supported by the Editions of SQL Server 2012. [Viitattu 2.4.2014]. Saatavissa: http://msdn.microsoft.com/en-us/library/cc645993.aspx.
12. Microsoft Corporation. 2014. Books Online for SQL Server 2012. [Viitattu 2.4.2014]. Saatavissa: http://technet.microsoft.com/en-us/library/ms130214.aspx.
13. International Business Machines Corporation (IBM). 2014. Framework Manager User Guide 10.1.1. [Viitattu 2.4.2014]. Saatavissa: http://pic.dhe.ibm.com/infocenter/cbi/v10r1m1/index.jsp?topic=%2Fcom.ibm.swg.ba.cognos.ug_fm.10.1.1.doc%2Ft_frameworkmanager.html.

14. International Business Machines Corporation (IBM). 2010. IBM Cognos Transformer User Guide. [Viitattu 2.4.2014]. Saatavissa: http://public.dhe.ibm.com/software/data/cognos/documentation/docs/en/10.1.0/ug_cogtr.pdf.
15. International Business Machines Corporation (IBM). 2014. Analysis Studio User Guide 10.2.1. [Viitattu 13.4.2014]. Saatavissa: http://www-01.ibm.com/support/knowledgecenter/SSEP7J_10.2.1/com.ibm.swg.ba.cognos.ug_cr_pps.10.2.1.doc/c_id_and_intro.html%23id_and_intro